МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КІБЕРБЕЗПЕКИ, КОМП'ЮТЕРНОЇ ТА ПРОГРАМНОЇ ІНЖЕНЕРІЇ
Кафедра комп'ютерних інформаційних технологій

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
Савченко А.С.
«___»_____2020 р.

ДИПЛОМНА РОБОТА

(ПОЯСНЮВАЛЬНА ЗАПИСКА) ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ "МАГІСТР"

ЗА СПЕЦІАЛІЗАЦІЄЮ "ІНФОРМАЦІЙНІ УПРАВЛЯЮЧІ СИСТЕМИ ТА ТЕХНОЛОГІЇ (ЗА ГАЛУЗЯМИ)"

Тема: «Екосистема Web-сайту для підприємства пасажирських перевезень»

Виконавець: Федорчук Дмитро Ігорович

Керівник: к.т.н., доцент Холявкіна Тетяна Володимирівна

Нормоконтролер: _____ Райчев І.Е.

Київ 2020

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки, комп'ютерної та програмної інженерії

Кафедра Комп'ютерних інформаційних технологій

Галузь знань, спеціальність, спеціалізація: 12 "Інформаційні технології", 122 "Комп'ютерні науки", "Інформаційні управляючі системи та технології (за галузями)"

ЗАТВЕРДЖУЮ
Завідувач кафедри
Савченко А.С.
"_____"_____ 2020 р.

ЗАВДАННЯ

на виконання дипломного проекту студента

Федорчука Дмитра Ігоровича
(прізвище, ім'я, по батькові)

1. **Тема роботи:** «Екосистема Web-сайту для підприємства пасажирських перевезень» затверджена наказом ректора №1891/ст. від 02.10.2020р..
2. **Термін виконання роботи:** з 05.10.2020р. по 31.12.2020р.
3. **Вихідні дані до роботи:** мінімальний об'єм оперативної пам'яті (RAM) – 1 GB; будь-який установлений сучасний браузер, сервер з мінімальним об'ємом оперативної пам'яті 1,5 GB та процесором 2 ядра.
4. **Зміст пояснювальної записки:** вступ, основи створення сайтів, огляд і порівняння локальних серверів, компонування архітектури екосистеми, висновок.
5. **Перелік обов'язкового графічного матеріалу:** адміністративні панелі систем управління контентом, види карт сайту, макет веб-додатку, компонування клієнтської бази.

6. Календарний план-графік

№ з/п	Завдання	Термін виконання	Підпис керівника
1.	Проаналізувати літературу та джерела за темою дипломного проекту.	05.10.20 – 10.10.20р.	
2.	Розроблення та затвердження плану дипломного проекту.	11.10.20 – 15.10.20р.	
3.	Провести консультації з науковим керівником щодо створення першого розділу.	16.10.20 – 18.10.20р.	
4.	Розробка розділу 1: Основи Створення Сайтів	19.10.20 – 29.10.20р.	
5.	Розробка розділу 2: Архітектура Веб-Сайту	30.10.20 – 10.11.20р.	
6.	Розробка розділу 3: Основи Php, Javascript,Css And Html5	11.11.20 – 21.11.20р.	
7.	Висновки та оформлення пояснювальної записки дипломного проекту.	22.11.20 – 02.12.20р.	
8.	Підписання необхідних документів у встановленому порядку.	03.12.20-10.12.20	
9.	Підготовка до захисту та попередній захист дипломного проекту на випусковій кафедрі дипломного проекту	11.12.20 – 21.12.20	

7. Дата видачі завдання: 05.10.2020р.

Керівник дипломного проекту _____ Холявкіна Т.В.
(підпис керівника) (П.І.Б.)

Завдання прийняв до виконання _____ Федорчук Д.І.
(підпис випускника) (П.І.Б.)

РЕФЕРАТ

Пояснювальна записка до дипломної роботи роботи «Екосистема Web-сайту для підприємства пасажирських перевезень» викладена на 92 сторінках, містить 21 рисунок, 5 наукових джерел.

Ключові слова: LANDING PAGE, ЦІЛЬОВА СТОРІНКА, ОДНОСТОРІНКОВИЙ САЙТ, СУЧАСНІ ТЕХНОЛОГІЇ СТВОРЕННЯ САЙТІВ, ЕКОСИСТЕМА.

Об'єкт дослідження: односторінкові веб-додатки Single Page Application.

Предмет дослідження: створення односторінкового веб-додатку і екосистеми за допомогою JavaScript та PHP.

Мета роботи: створити сучасний веб-додаток та екосистему для ефективного ведення клієнтів за допомогою JavaScript та PHP.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ, ТЕРМІНІВ

HTML - HyperText Markup Language

CSS - Cascading Style Sheets

JS – JavaScript

PHP - Hypertext Preprocessor (old Personal Home Page)

SQL - Structured query language

ВСТУП

Близько 71% малих підприємств мають веб-сайти, і майже всі великі підприємства мають систему автоматизації стратегій взаємодії з клієнтами та партнерами.

Присутність в Інтернеті може принести значну користь бізнесу. Розглянемо деякі з основних переваг веб-сайту:

- Ви можете ефективно спілкуватися зі своїми клієнтами. Ваш веб-сайт може бути універсальним рішенням для клієнтів і потенційних клієнтів, які можуть знайти те, що їм потрібно.
- У вас більше довіри. Бізнес, який не має професійного веб-сайту, як правило, не сприймається клієнтами дуже серйозно.
- Ви можете залучати клієнтів 24/7. Наявність веб-сайту означає, що ваші клієнти можуть знайти вас в будь-який час і в будь-якому місці.
- Веб-сайт – це маркетинговий інструмент, який працює сам по собі. Вам не потрібно сидіти за комп'ютером, щоб допомогти людям знайти ваш сайт. Як тільки веб-сайт буде запущений, він виконає свою роботу.

Сторінки сайтів - це набір текстових файлів, розмічених мовою HTML. Ці файли, будучи завантаженими відвідувачем на його комп'ютер, розуміються і обробляються браузером і виводяться на засіб відображення користувача (монітор, екран КПК, принтер або синтезатор мови). Мова HTML дозволяє форматувати текст, розрізняти в ньому функціональні елементи, створювати гіпертекстові посилання (гіперпосилання) і вставляти в сторінку зображення, звукозаписи і інші мультимедійні елементи. Відображення сторінки можна змінити додаванням стилів на мові CSS, що дозволяє централізувати в певному файлі всі елементи форматування (розмір і колір заголовних букв 2-го рівня, розмір і вид блоку вставки і інше) або сценаріїв на мові JavaScript, за допомогою якого є можливість переглядати сторінки з подіями або діями.

Сторінки сайтів можуть бути простим статичним набором файлів або створюватися спеціальною комп'ютерною програмою на сервері. Вони можуть бути або зроблені на замовлення для окремого сайту, або бути готовим продуктом, розрахованим на певний клас сайтів. Деякі з них можуть забезпечити власнику сайту можливість гнучкої настройки структуризації і виведення інформації на веб-сайті. Такі керуючі програми називаються системами управління змістом (CMS).

Сайти можуть містити підрозділи, орієнтовані цілком на ту чи іншу аудиторію. У цьому випадку такі розділи називають версіями сайту. Аудиторія може відрізнятися по виду використовуваного обладнання, по використовуваному мови аудиторії. Наприклад, відомі так звані мобільні версії сайту, призначені для роботи з ними з використанням смартфона. Сайти можуть мати мовні версії (російськомовна, англомовна і інші).

Більшість людей будуть сканувати Інтернет в пошуках товару або послуги перед покупкою, щоб спочатку перевірити надійність. Коли ви надаєте відмінну послугу, позитивний словесний обмін думками про ваш бізнес, ймовірно, буде поширюватися. Таким чином, з'являється більше повторних і нових замовлень. Люди зазвичай довіряють бізнесу після того, як з ним попрацювали. Використовуючи свій сайт, можна постійно обслуговувати споживачів в Інтернеті і підвищувати свій авторитет як підприємця.

Ефективність сайту залежить від правильного підходу до аналізу бізнесу, підборі виду сайту та визначенні цільової аудиторії. Також створюючи веб-сайт, потрібно чітко розуміти свою головну мету.

Не слід забувати і про його зовнішній вигляд. Чому веб-сайт повинен добре виглядати? Чому це має виглядати професійно? Тому що, як вивіска

над магазином в торговому центрі, веб-сайт відображає і бізнес. Потрібно зробити так, щоб сайт виглядав на вищому рівні.

Існує безліч сайтів, які є значущими ресурсами. На цих ресурсах можуть розташовуватися персональні дані користувачів (наприклад, особисте листування, адреси, телефони) або фінансова інформація (наприклад, банківські сайти). Злом таких ресурсів може спричинити як прямі грошові збитки (наприклад, зловмисник може перерахувати гроші з чужого рахунку на свій власний), так і непрямі, пов'язані з поширенням конфіденційної інформації або просто зловмисник може зіпсувати вміст сайту. Для багатьох сайтів важливо забезпечити певний рівень безпеки. Необхідний рівень безпеки багато в чому залежить від церкви, розміщеної на сайті інформації.

Отже, виходячи з цього, можна сказати, що сайт – це необхідність для підприємства в сучасному світі. Саме з цієї причини я вирішив зробити веб-додаток в екосистемі для нового підприємства пасажирських перевезень.

РОЗДІЛ 1

ОСНОВИ СТВОРЕННЯ САЙТІВ

1.1. Основна потреба в веб-сайті.

Сьогодні власний веб-сайт має майже кожна організація, оскільки використання сучасних інформаційних технологій – необхідний чинник існування, що дозволяє збільшити кількість клієнтів та їх рівень задоволення завдяки швидкому доступу до необхідної інформації.

Створення і розробка сайтів включає:

1. Затвердження первинного технічного завдання на розробку сайта.

2. Визначення структурної схеми сайту - розташування розділів, контента і навігації.

3. Веб-дизайн - створення графічних елементів макету сайту, стилів і елементів навігації.

4. Розробка програмного коду, модулів, бази даних і інших елементів сайту необхідних в проекті.

5. Тестування і розміщення сайту в мережі Інтернет.

1.2. Поняття веб-сайту.

Веб-сайт – це одна або більше логічно зав’язаних між собою веб-сторінок. Зазвичай сайт в інтернеті представляє масив даних, що мають унікальну адресу. Доступ до сайту відбувається по протоколу HTTP.

Сторінки сайту – це набір текстових файлів, розмічених на мові HTML. HTML (від англ. HyperText Markup Language) – стандартизована мова розмітки веб-сторінок.

Після загрузки html файлів на комп’ютер, вони обробляються браузером і виводяться на екран користувача.

Мова HTML дозволяє редагувати текст, виводити в ньому функціональні елементи, створювати гіпертекстові посилання, вставляти зображення, звукові записи, а також мультимедійні елементи. Відображення сторінки можливо змінити за допомогою мови CSS, тобто додаванням стилів на мові CSS. Це дає змогу централізувати в одному файлі всі елементи форматування (розмір, колір, вид блоку та інше) або сценаріїв на мові JavaScript, за допомогою якої ми маємо можливість створювати сторінки з подіями.

Сторінки сайтів можуть бути як простим статичним набором файлів, так і створюватись спеціальною комп’ютерною програмою на сервері.

Сайти можуть містити підрозділи, орієнтовані цілком на ту чи іншу аудиторію. В такому випадку розділи називають версіями сайту. Аудиторія може різнитись за видом використовуваного обладнання, а також за використовуваної мови аудиторії. Наприклад, відомі мобільні версії сайту, які створюються для роботи з ними з використанням смартфону. Також сайти можуть мати різні язикові версії (українська, англійська та інші).

1.3. Класифікація веб-сайтів.

Всі веб-сайти можна класифікувати:

- По доступності сервісів.
- По змісту.
- По фізичному розташуванню.
- По схемі представлення інформації, її об’єму і категорії вирішення задач.
- По відношенню до користувача.

По доступності сервісів:

- Відкриті – всі сервіси доступні для любих користувачів.

- Напіввідкриті – для доступу необхідно зареєструватись.
- Закриті – повністю закриті службові сайти організацій, особисті сайти приватних осіб.

По змісту:

- Статичні – користувачу видаються файли у вигляді, в якому вони зберігаються на сервері.
- Динамічні – зміст генерується спеціальними скриптами на основі інших даних.

По фізичному розташуванню:

- Зовнішні сайти – в мережі Інтернет.
- Локальні сайти – доступні тільки в локальній мережі.

По схемі представлення інформації, її об'єму і категорії вирішення задач:

✓ Інтернет-представництва власників бізнесу:

- Сайт-візитка – містить загальні дані про власника сайту.
- Корпоративний сайт – містить повну інформацію про компанію, послуги, продукцію.
- Каталог продукції – в каталозі присутній детальний опис товарів або послуг,
- Інтернет-магазин – веб-сайт з каталогом продукції, за допомогою якого клієнт може замовити потрібні йому товари. Для таких сайтів використовуються різноманітні системи розрахунків: від пересилки товару накладеним платежем, до розрахунків за допомогою пластикових карток.
- Промо-сайт – сайт про конкретну торгівельну марку або продукт, на таких сайтах розміщується інформація про бренд, різноманітні рекламні акції, тощо.
- Сайт-квест – Інтернет ресурс, на якому організовано змагання по розгадуванню послідовності взаємопов'язаних між собою логічних загадок.

✓ По відношенню до користувача:

- Сайт, що залучає користувача.
- Байдужий до відвідувача.

1.4. Етапи розробки веб-сайта.

Існують такі основні етапи розробки веб-сайту:

- Постановка цілей і задач.
- Створення макету дизайну сайту.
- Верстка.
- Програмування.
- Наповнення контентом.
- Тестування.

1.4.1. Постановка цілей і задач.

Перед тим, як переходити до етапу створення макету дизайну веб-сайту потрібно поставити ціль і задачу, яку майбутній сайт буде виконувати. Після того, як поставлені завдання, визначається цільова аудиторія. Важливо зрозуміти для кого створюється ресурс.

1.4.2. Створення макету дизайну сайту.

Дизайн - це те, що бачить відвідувач в першу чергу, оцінює його і приймає рішення залишитися на сторінці або закрити вкладку браузера.

Для якісного дизайну потрібно намалювати кнопки, банери та інші графічні елементи. Іншими словами, той прототип, який був створений на першому етапі розробки сайту, отримує естетичний зовнішній вигляд.

Важливо відзначити, що потрібно створювати дизайн не кожної сторінки, а шаблони кількох основних, використовуючи тенденції веб-дизайну.

Титульна сторінка (головна) будь-якого сайту повинна максимально інформативно і в стислому об'ємі відображати необхідну користувачеві

інформацію про сайт. На головній сторінці необхідно помістити логотип веб-сайту, основне меню сайту (для навігації по його структурі), форму аутентифікації (входу зареєстрованих користувачів), реєстраційне посилання (реєстрація нових клієнтів).

1.4.3. Верстка.

Наступним етапом є переведення дизайну сайту в робочий проект за допомогою мови HTML. Ресурс отримує життя, стає динамічним, все кнопки працюють. Ресурс стає кросбраузерності і правильно відображається у всіх існуючих інтернет-браузерах. На цьому ж етапі створюються стилі CSS.

На цьому ж етапі верстаються категорії і підкатегорії, контент, який буде розміщений на сторінці.

На завершальному етапі верстки розробляється адаптивна версія сайту.

1.4.4. Програмування.

На даному етапі потрібно провести розробку складних функцій: додавання карти, робота з базою даних, робота калькулятору. Тобто створення того функціоналу, що не входить в стандартну верстку веб-сайту.

1.4.5. Наповнення контентом.

Передостаннім етапом створення веб-сайту – є наповнення його інформаційним контентом. Розміщується вся інформація, яку зможе побачити і прочитати користувач.

1.4.6. Тестування.

Найважливішим етапом в процесі створення веб-сайту – є його тестування. Під час тестування необхідно провести моніторинг функціональності сайту для виявлення помилок і їх подальшого усунення.

1.5. Мова програмування PHP.

PHP - скриптова мова загального призначення, інтенсивно застосовується для розробки веб-додатків. В даний час підтримується переважною більшістю хостинг-провайдерів і є одним з лідерів серед мов, що застосовуються для створення динамічних веб-сайтів.

PHP-скрипти зазвичай обробляються інтерпретатором в порядку, що забезпечує кроссплатформенність розробленого додатка:

- Лексичний аналіз вихідного коду і генерація лексем.
- Синтаксичний аналіз отриманих лексем.
- Генерація байт-коду.
- Виконання байт-коду інтерпретатором (без створення виконуваного файлу).

Для збільшення швидкодії додатків можливе використання спеціального програмного забезпечення, так званих акселераторів. Принцип їх роботи полягає в кешуванні одного разу згенерованого байт-коду в пам'яті і / або на диску, таким чином, з процесу роботи програми виключаються етапи 1-3, що в загальному випадку веде до значного прискорення роботи.

Важливою особливістю є те, що розробнику немає необхідності піклуватися про розподіл і звільнення пам'яті. Ядро PHP реалізує засоби для автоматичного керування пам'яттю; вся виділена пам'ять повертається системі після завершення роботи скрипта.

Інтерпретатор складається з ядра і модулів, «розширень», що представляють собою динамічні бібліотеки. Розширення дозволяють доповнити базові можливості мови, надаючи можливості для роботи з базами даних, сокетами, динамічною графікою, криптографічними бібліотеками, документами формату PDF і тому подібним. Будь-який бажаючий може розробити своє власне розширення і підключити його. Існує величезна кількість розширень, як стандартних, так і створених сторонніми компаніями

і ентузіастами, однак в стандартну поставку входить лише кілька десятків добре зарекомендували себе. Безліч розширень є в репозиторії PECL.

1.6. Мова програмування JavaScript.

JavaScript – мультіпарадігменна мова програмування. Підтримує об'єктно-орієнтована, імперативний і функціональний стилі.

JavaScript зазвичай використовується як вбудований мова для програмного доступу до об'єктів додатків. Найбільш широке застосування знаходить в браузерах як мова сценаріїв для додання інтерактивності веб-сторінок.

Основні архітектурні риси: динамічна типізація, слабка типізація, автоматичне керування пам'яттю, прототипне програмування, функції як об'єкти першого класу.

На JavaScript вплинули багато мов, при розробці була мета зробити мову схожим на Java. Мовою JavaScript не володіє будь-яка компанія або організація, що відрізняє його від ряду мов програмування, використовуваних в веб-розробці.

JavaScript є об'єктно-орієнтованою мовою, але що використовується в мові прототипирование обумовлює відмінності в роботі з об'єктами в порівнянні з традиційними клас-орієнтованими мовами. Крім того, JavaScript має ряд властивостей, властивих функціональним мовам, - функції як об'єкти першого класу, об'єкти як списки, каррінг, анонімні функції, замикання - що додає мові додаткову гнучкість.

Незважаючи на схожий з Сі синтаксис, JavaScript в порівнянні з мовою Сі має корінні відмінності:

- Об'єкти з можливістю інтроспекції.
- Функції як об'єкти першого класу.
- Автоматичне приведення типів.
- Автоматичне прибирання сміття.
- Анонімні функції.

Якщо розглядати JavaScript в відмінних від браузера середовищах, то об'єктна модель браузера і об'єктна модель документа можуть не підтримуватися.

Об'єктну модель документа іноді розглядають як окрему від JavaScript сутність, що узгоджується з визначенням DOM як незалежного від мови інтерфейсу документа. На противагу цьому ряд авторів знаходить BOM і DOM тісно взаємопов'язаними.

JavaScript використовується в клієнтської частини веб-додатків: клієнт-серверних програм, в якому клієнтом є браузер, а сервером - веб-сервер, що мають розподілену між сервером і клієнтом логіку. Обмін інформацією в веб-додатках відбувається по мережі. Одним з переваг такого підходу є той факт, що клієнти не залежать від конкретної операційної системи користувача, тому веб-додатки є кросплатформенними сервісами.

1.7. Реляційна система управління базами даних.

На даний момент MySQL являє собою одну з найбільш надійних, швидких, якісних і відомих з усіх існуючих сучасних систем управління базами даних, яку дуже просто використовувати із сайтами. Основною причиною цього є її безкоштовне розповсюдження разом зі своїми вихідними кодами, інша причина - це те, що MySQL досить швидка СУБД.

Головна особливість роботи СУБД MySQL полягає в використанні мови структурованих запитів - SQL в ролі основного в управлінні базою даних, а саме: для створення або видалення таблиць в базі даних, здійснення вибірки з бази даних, для безпосереднього заповнення таблиць даними. MySQL має високу стійкість, високою швидкістю роботи, простотою в налаштуванні і використанні, вихідні коди сервера компілюються на безлічі платформ, тому СУБД MySQL є гідним рішенням для невеликих програм.

Основні можливості MySQL:

- Надає можливість одночасної роботи з базою даних необмеженому числу користувачів.
- Кількість рядків у таблицях може досягати 50 млн.
- Висока швидкість виконання команд користувачів.
- Проста і ефективна система безпеки.

В якості ще одного переваги перед іншими СУБД можна виділити те, що MySQL може працювати з мовою SQL в стандарті ANSI 92, а також має безліч встановлюються розширень до цього стандарту, яких не передбачено ні в жодній іншій системі управління базами даних.

Недоліком даної СУБД є відсутність підтримки вкладених запитів, типу SELECT * FROM first_table WHERE recordId IN (SELECT recordId FROM second_table). Також не реалізована підтримка транзакцій і не передбачена підтримка тригерів та збережених процедур.

1.8. Огляд і порівняння сучасних систем управління контентом.

На даний момент розроблено велику кількість готових систем керування вмістом контентом, як платних, так і безкоштовних. Всі ці системи можна розділити на три категорії:

- Створення сторінки за запитом користувача. Системи з цієї категорії працюють по шляху: «Модуль редагування → База даних → Модуль уявлення». Модуль уявлення виробляє безпосереднє наповнення вмісту сторінки за допомогою інформації з бази даних при запиті користувача. Для внесення змін до бази даних використовується модуль редагування. Недоліком даної категорії є те, що при кожному запиті користувача відбувається створення сторінки, що тягне навантаження на ресурси Web-сервера. Але завдяки використанню засобів кешування навантаження на сервер може бути значно знижена.

- Створення сторінок в режимі редагування. Системи з цієї категорії виступають в якості додатків для редагування коду Web-сторінки, за допомогою яких після внесення змін заново створюють набір статичних сторінок. Недоліком даної категорії є те, що при даному порядку створення сторінок втрачається інтерактивний зв'язок між користувачем і контентом сайту.

- Змішаний тип. Дана категорія об'єднує в собі переваги перших двох. Системи з цієї категорії можуть працювати і по шляху кешування, тобто коли модуль уявлення створює Web-сторінку один раз і при наступному запиті користувача вона набагато швидше завантажується за допомогою кешування. Оновлення кешу може відбуватися як автоматично, по закінченню певного періоду часу або при редагуванні певних розділів Web-сайту, так і вручну за допомогою спеціальних команд. Інший підхід полягає в збереженні певних інформаційних блоків на етапі внесення змін до структури Web-сайту.

WordPress - одна з найбільш популярних безкоштовних CMS в усьому світі. Головним її призначенням є створення і реалізація сайту-блогу. CMS WordPress дуже просто і легко встановлюється, практично за пару кліків. Відразу необхідно відзначити велику кількість існуючих тим і шаблонів оформлення зовнішнього вигляду сайту. Створення проекту за допомогою даної CMS не вимагає спеціальних або додаткових знань, що є ідеальним варіантом для тих, хто вперше стикається зі створенням сайту. CMS WordPress має дуже інтуїтивно зрозумілий інтерфейс панелі адміністратора.

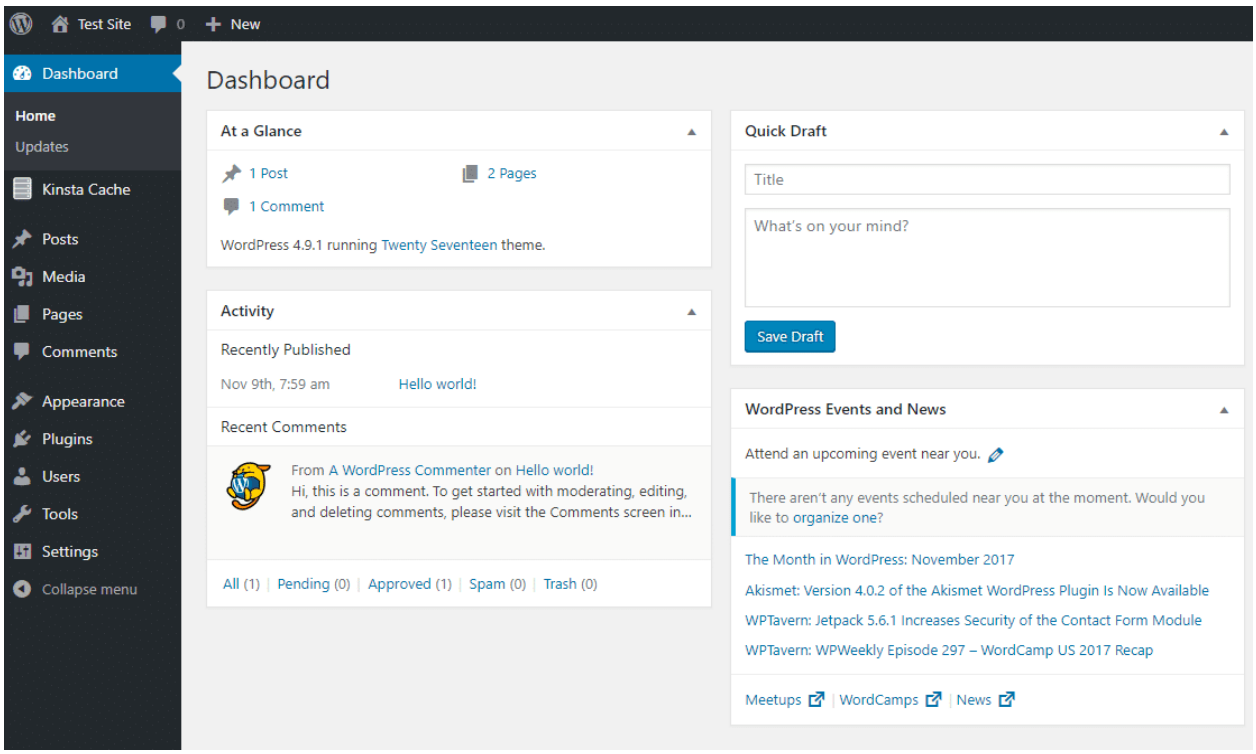

Рис. 1.1. Адміністративна панель WordPress

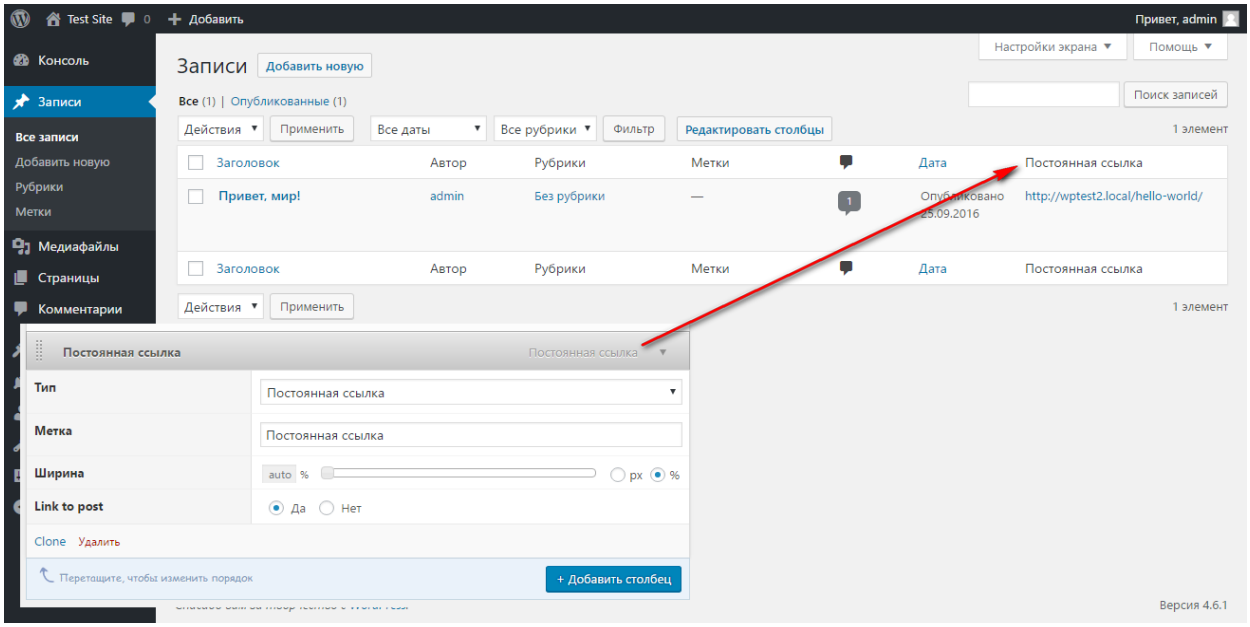

Рис. 1.2. Адміністративна панель WordPress

Основне призначення WordPress - створення і ведення сайтів блогів, однак варто відзначити, що також можливе створення на даній CMS і різних журналів, магазинів, каталогів статей або просто сайтів також зручно і швидко. Додавання різних функціональних можливостей стало доступним

завдяки тисячам існуючих доповнень, які написані спеціально для даної CMS і призначених для значного розширення можливостей.

Як недоліки даної системи можна віднести, що вона спеціалізується на створенні і супроводі блогів, а створення сайтів з іншою категорією може виявитися значно складнішим завданням в порівнянні з іншими, більш універсальними системами.

Joomla є дуже популярну гнучку безкоштовну CMS. На базі даної системи побудовано величезну кількість як невеликих, так і дуже великих проектів, які мають величезні функціональні можливості. Установка системи Joomla є автоматизованою і дуже простий. Є величезна кількість всіляких тем оформлення, які легко встановлюються з панелі адміністратора. Всі необхідні настройки також виробляються безпосередньо з панелі адміністратора, яка володіє простим і інтуїтивно зрозумілим інтерфейсом.

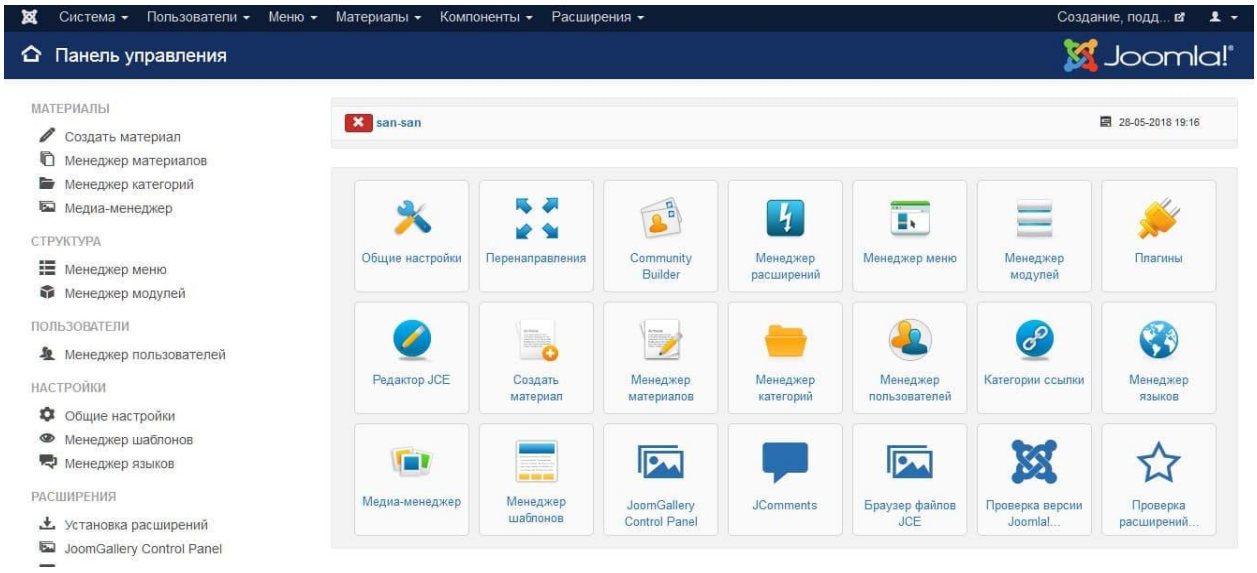

Рис. 1.3. Адміністративна панель Joomla

Дана система управління контентом також має величезну кількість розширень, які легко і просто встановлюються з панелі адміністратора. Однак необхідно відзначити той факт, що для створення якісного сайту з використанням Joomla будуть потрібні значні знання мови розмітки HTML і таблиці стилів CSS. З адмініструванням проекту не пов'язано великих проблем. Всі необхідні дії реалізовуються з панелі

адміністратора. Для реалізації додавання нових матеріалів використовується візуальний редактор статей і завантажувач зображень. Joomla є дуже гнучкою завдяки великій кількості існуючих розширень, за допомогою яких можна створювати різні новинні портали, відео та фото галереї, каталоги нерухомості, магазини, мультимовні сайти, соціальні мережі, дошки оголошень та інші тематичні сайти. До недоліків даної системи можна віднести те, що, незважаючи на всі існуючі зручності, для створення проекту знадобляться значні знання в області CSS і HTML. Також необхідно відзначити, що дуже ретельно необхідно вибирати розширення і віддавати перевагу тим, які є найбільш популярними, якісними і мінімально завантажувати сервер.

Drupal є також популярною CMS, яка призначена для створення сайтів-порталів. Установка даної системи дуже проста і не викликає проблем. Але варто зазначити, що новачкам буде потрібно якийсь час, щоб розібратися і звикнути до даної CMS.

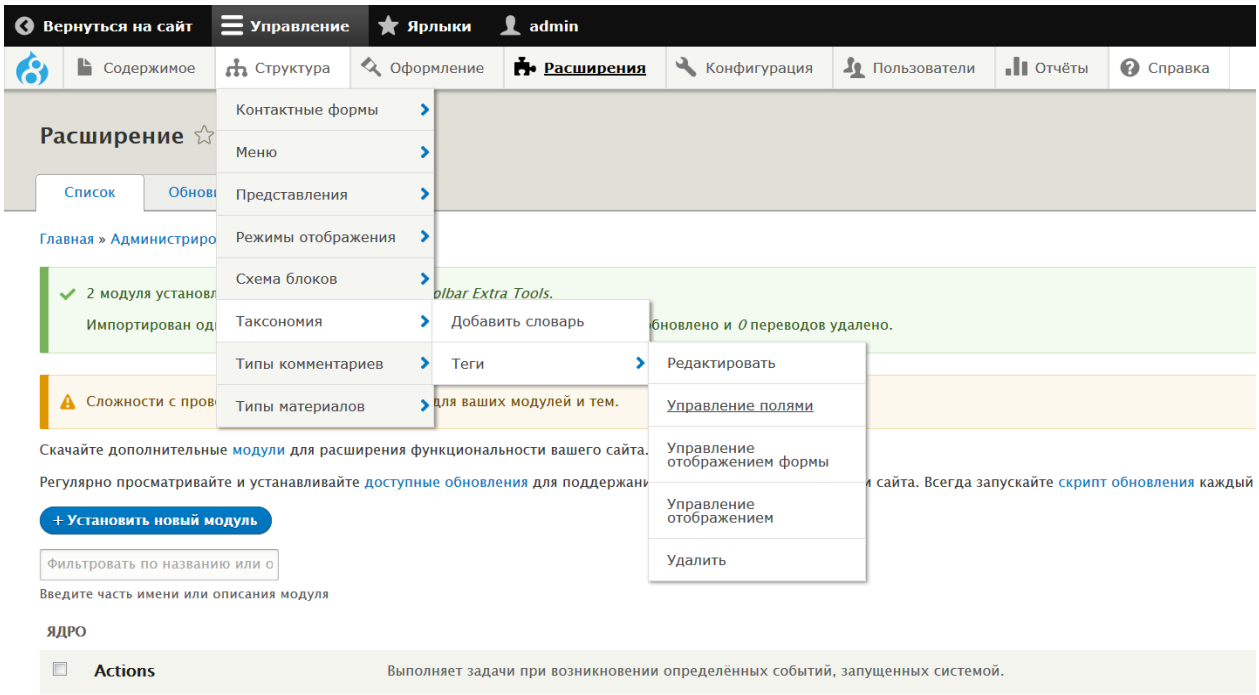

Рис. 1.4. Адміністративна панель Drupal

Для того, щоб інтегрувати шаблони, знадобляться певні початкові знання в області програмування. Адміністрування системою не викликає особливих труднощів, але вимагає конкретних знань. Для додавання матеріалів на сайт можна використовувати візуальний редактор, який необхідно встановлювати додатково, так як він не йде в ядрі даної CMS.

Вставка картинок чи інших зображень також не передбачена в базовому пакеті даної CMS, однак дана проблема також вирішується за допомогою використання додаткових компонентів. Необхідно відзначити, що Drupal є гнучкою системою, яка так само, як і інші розглянуті системи, має велику кількість різних доповнень, які дозволяють створювати сайти різної тематики - соціальні мережі, галереї, форуми, сайти новин. Недоліком даної системи, перш за все, є те, що не кожен зможе швидко створити якісний сайт.

Система управління контентом DLE (Datalife Engine) найчастіше використовується для розробки різних новинних ресурсів. Однак зараз випущено велику кількість всіляких доповнень і розширень, які дозволяють DLE використовувати для вирішення будь-яких завдань. Дана CMS вважається дуже якісної і її знає практично кожен Web-майстер. За допомогою даної системи розроблено велику кількість різнопланових ресурсів. DLE від розглянутих вище систем управління відрізняє той факт, що вона не є безкоштовною, проте все вкладення будуть повністю виправдані, так як дана CMS здатна запропонувати дуже велика кількість переваг перед іншими системами. DLE має дуже зрозумілим і зручним інтерфейсом. Ніяка інша система не є такою ж простою, як DLE. Інтерфейс CMS надзвичайно простий і зручний для сприйняття і не здатний викликати зайвих питань. Для системи даного рівня інтерфейс є найдоступнішим.

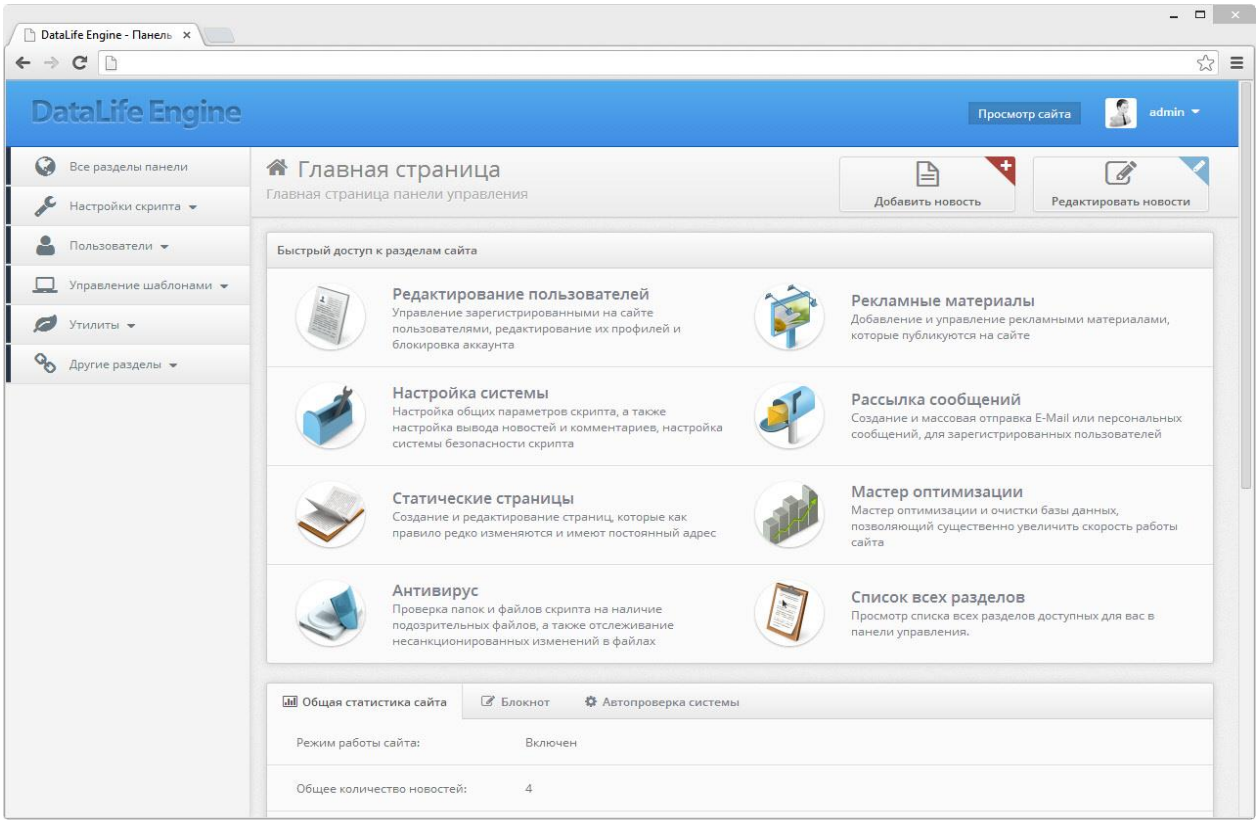

Рис. 1.5. Адміністративна панель DLE

Важливим також є той факт, що дана система управління ставить мінімальні вимоги до хостингам, що дозволяє продуктивності сайту залишатися на високому рівні при будь-яких умовах.

Саме завдяки цій характеристиці DLE рекомендується використовувати для створення новинних проектів, так як сайт може спокійно приймати тисячі відвідувачів одночасно. Що стосується функціональності, то тут також все на високому рівні. Для повноцінної роботи з новинами тут присутні всі необхідні функції, а також додаткові розширення, які гарантують саму оперативну публікацію статей в поєднанні з дуже простим і легким редагуванням даних.

Завдяки розробленим шаблонами, модулями і інструментам системи управління DLE можна створювати різні проекти: інтернет-магазин, соціальну мережу або форум. Але найкраще за допомогою даної CMS створювати новинні портали. До недоліків даної системи можна віднести те, що дана система управління контентом є платною, хоча в безкоштовній

версії доступна велика кількість функцій, але є обмеження на кількість розміщуваних матеріалів.

1.9. Огляд і порівняння локальних серверів.

Локальний сервер - це набір програм, які створюють аналог хостингу безпосередньо на вашому комп'ютері і дозволяють створювати і тестувати динамічні сайти з використанням серверних мов програмування і бази даних.

Локальний сервер AMPPS.

Локальний сервер AMPPS - це збірник програм, які створюють цілу бібліотеку для роботи веб майстра. Кожна програма можна встановити, протестувати і використовувати в своїх цілях, причому робити це дуже швидко.

Ця збірка має свої плюси і переваги перед іншими. Варто відзначити підтримку поширених CMS, веб-форумів, блог-платформ, фотогалерей, дощок оголошень і інше. Кількість доступних додатків налічує понад 200 штук.

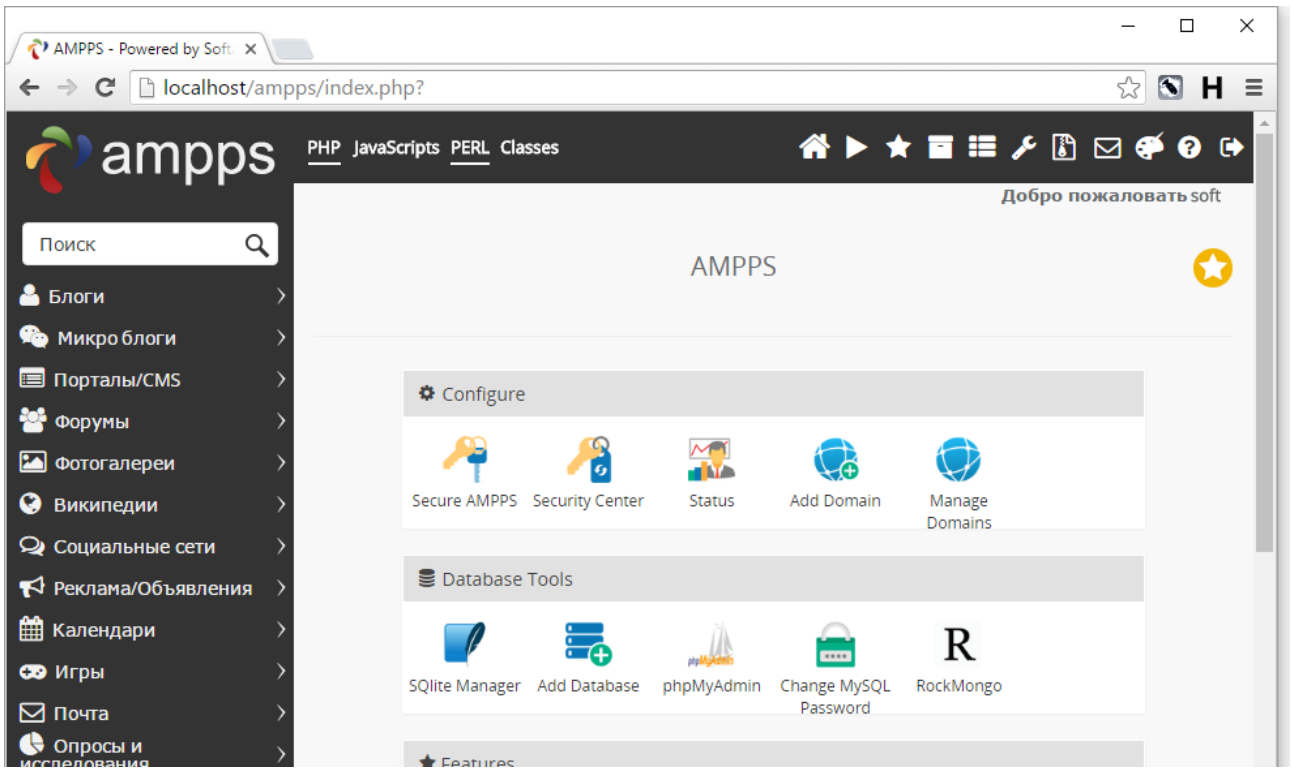

Рис. 1.6. Адміністративна панель AMPPS

З мінусів можна відзначити частковий російський переклад, який присутній тільки в каталозі скриптів. Ще одним мінусом є відсутність портативності.

Локальний сервер AppServ.

Локальний сервер AppServ - це набір додатків, які встановлюються як сервіси. Як недолік можна виділити відсутність керуючої програми, тільки сторінка лаконічного дизайну і посилання на PhpMyAdmin. За своєю суттю AppServ є графічним монтажником Apache, PHP, MySQL і більш нічим.

Локальний сервер WampServer.

Локальний сервер WampServer - це безкоштовна збірка платформи для Web-розробки під операційною системою Windows. До складу цієї збірки включено все необхідне програмне забезпечення для створення динамічних Web-сайтів, а саме: мову програмування PHP, СУБД MySQL і інше

допоміжне ПО. Основні переваги WampServer: легка установка, невеликий розмір дистрибутива, стабільність роботи. Основним недоліків WampServer є те, що WampServer потребує попереднього встановлення.

Локальний сервер Denwer.

Головна особливість Denwer - це комфортні умови при віддаленій роботі відразу над декількома незалежними проектами і можливість розміщення на Flash-накопичувачі. Крім того, емулятор Web-сервера Denwer може бути використаний для створення власного хостингу, а так само при створенні баз даних для Web-сайтів, використовуючи при цьому мову SQL.

ВИСНОВОК ДО РОЗДІЛУ 1

Зараз існує велика кількість систем управління контентом, які дозволяють користувачам швидко і без зайвих дій змінювати наповнення сайту та здійснювати контроль додатку. Якщо немає потреби в повністю кастомізованому дизайні і серверній частині, можна використовувати існуючи системи управління контентом, наприклад, WordPress або Joomla.

Основні із них:

- WordPress - одна з найбільш популярних безкоштовних CMS в усьому світі. Головним її призначенням є створення і реалізація сайту-блогу. CMS WordPress дуже просто і легко встановлюється, практично за пару кліків. Відразу необхідно відзначити велику кількість існуючих тим і шаблонів оформлення зовнішнього вигляду сайту.
- Joomla є дуже популярну гнучку безкоштовну CMS. На базі даної системи побудовано величезну кількість як невеликих, так і дуже великих проектів, які мають величезні функціональні можливості.
- Drupal є також популярною CMS, яка призначена для створення сайтів-порталів. Установка даної системи дуже проста і не викликає проблем. Але варто зазначити, що новачкам буде потрібно якийсь час, щоб розібратися і звикнути до даної CMS.
- Система управління контентом DLE (Datalife Engine) найчастіше використовується для розробки різних новинних ресурсів. Однак зараз випущено велику кількість всіляких доповнень і розширень, які дозволяють DLE використовувати для вирішення будь-яких завдань. Дана CMS вважається дуже якісної і її знає практично кожен Web-майстер.

РОЗДІЛ 2

АРХІТЕКТУРА ВЕБ-САЙТУ

2.1. Що таке архітектура сайту?

Термін «архітектура сайту» охоплює загальний макет і дизайн вашого сайту, а також дрібні деталі; компоненти, про які багато людей, що запускають свій перший веб-сайт, не дуже замислюються, але точно знають.

Кожен веб-сайт має робити швидко і чітко наступне:

- Повідомляти, що продається на сайті чи пропонується
- Допомагати робити легку покупку або надавати доступ до товарів
- Надайте зручну контактну інформацію та посилання на облікові записи соціальних мереж власників сайту
- Легко підписатися чи відписатися на список розсилки електронної пошти

Отже, перш ніж ми обговоримо що-небудь, що пов'язано з контентом, навігацією, сортуванням карток і всіма іншими життєво-важливими елементами архітектури веб-сайту, необхідно зробити наступне:

- Визначити цілі створення сайту
- Розуміння користувача або клієнта
- Визначити мету веб-додатку
- Що сайт пропонує (послугу, товар чи інше)

2.2. Основна інформація

Основу будь-якого сайту створює його контент. Контент — інформаційний текст, зображення, відео або що-небудь ще, що може

знадобитися донести до користувача сайту. Перед створенням сайту і його дизайну необхідно визначити його тематику і його наповнення. Тільки після цього підбирається найбільш вдалий дизайн.

Тепер, коли є розуміння про наповнення сайту і його призначення, необхідно структурувати все. В цьому допоможуть основні компоненти, необхідні кожному веб-сайту:

- Домашня сторінка
- Зміст
- Сторінка «Про власника»
- Сторінка з контактною інформацією
- Цільові сторінки
- Навігація
- Оптимізація для мобільних пристроїв
- Привабливий дизайн

2.3. Домашня сторінка

Домашня сторінка - це один з найважливіших компонентів вашого сайту, так як саме вона презентує все, що пропонує сам сайт.

Наповнення головної сторінки буде варіюватися в залежності від того, для чого призначений цей веб-сайт, але в першу чергу на головній сторінці має бути відразу ясно, що ви продаєте або просуваєте. Вам потрібно звернутися до тих, хто тільки що зайшов на ваш сайт, і спонукати їх зробити конверсію.

На відміну від цільової сторінки домашня сторінка буде служити декільком цілям і залучати різні аудиторії. Тут головне - баланс. Вам потрібно донести свою точку зору, не перевантажуючи клієнта, але при цьому показати все, що ви можете запропонувати. Для цього існують наступні поради:

- Ясний і короткий текст, заголовок.

- Зображення чи відео.
- Відгуки.
- Проста у використанні навігація.

Домашня сторінка повинна містити багато посилань, наприклад:

- Найважливіші продукти
- Промо акції
- Пропозиції за змістом, наприклад електронна книга або технічний документ
- Блог
- Інші сторінки сайту, що допоможуть користувачу визначитись у виборі

Також на головній сторінці необхідно розмістити те, що ви хочете, щоб в першу чергу побачив той, хто вводить URL.

2.4. Ваш контент

Тип вмісту, який ви в кінцевому підсумку вирішите розмістити на своєму веб-сайті, буде багато в чому залежати від його намірів. Ключовим моментом є створення конкретних цільових сторінок. В якості додаткового бонусу добре структурований веб-сайт з більшою ймовірністю отримає більш високий рейтинг в пошукових системах.

На головній сторінці вашого веб-сайту електронної комерції ваш клієнт повинен мати можливість перейти до:

- Категорії
- Підкатегорії
- сторінки продуктів

Кожен продукт, який ви продаєте чи пропонуєте, повинен бути в одній із категорії, а потім розділений на більш конкретні підкатегорії. При

визначенні того, як маркувати категорії, безумовно пам'ятайте про дослідження SEO та ключові слова.

Ключові плагіни для сайту електронної комерції включають:

- Плагін кошика покупок
- Плагін підтримки чату
- Плагін відгуків

Також необхідно подбати про правильний і детальний опис продуктів. Не менш важливо звернути увагу на граматичну складову всього контенту на сайті. Це корисно не тільки для вашого потенційного клієнта, а й для рейтингу SEO.

Якщо сайт пропонує професійні послуги, то потрібно продумати всі дії, які кінцевий користувач може і хоче виконати на сайті. Для цього необхідні категорії, що добре описують послугу:

- Сервіси
- Пов'язані послуги
- Меню
- Ціноутворення
- Спеціальні пропозиції та ін.

Ключові плагіни, які можуть знадобитися для сайту послуг, включають:

- Плагін кошика покупок
- Плагін для запису на прийом

Якщо говорити про блог-сайт, звернути увагу необхідно на кілька ключових компонентів:

- Категорії
- Теги
- Посилання на схожі повідомлення
- Послуги / Товари / Найміть мене

- Філії
- Сторінка підписок

2.5. SEO

Неможливо говорити про архітектуру веб-сайту, не згадавши про важливість SEO. Щоб з’ясувати, який контент потрібен і як його організувати — це з'ясувати, які ключові слова використовує цільова аудиторія при пошуку продуктів чи послуг, схожих на той, що пропонує веб-сайт.

Дослідження ключових слів

Виконуючи дослідження ключових слів і оцінюючи обсяг пошукових даних, можливо досить легко визначити, які категорії, підкатегорії і навіть фільтри необхідні для конкретного сайту. Необхідно використовувати такі інструменти, як Google Keyword Planner і SEMrush. Ці індикатори також можуть використовуватися для визначення того, чи слід створювати сторінку окремо або складати в іншу.

Розумні URL

Тепер, коли розібралися з розбивкою за категоріями і підкатегоріями, URL-адреси повинні це відображати. По суті, він повинен імітувати навігацію по структурі вашого сайту.

2.6. Сторінка «Про власника»

Коли справа доходить до ефективної сторінки «Про нас», ключовим моментом є досягнення балансу. Не потрібно перевантажувати користувача занадто великою кількістю інформації, але надзвичайно важливо проінформувати відвідувача і запевнити його в тому, що власник надійний і його продукція — це те що відвідувач шукав.

Також необхідно розповісти історію компанії і описати продукт чи послугу. Це може бути коротка і швидка розповідь, в якій можливо

використати відео або навіть інфографіку для більш вражаючого і незабутнього враження.

Також на цій сторінці варто розповісти про команду проекту. Навіть якщо команда на даному етапі не велика. Це ще одна можливість привернути увагу користувача і його довіру.

2.7. Сторінка контактів

Важливо мати кілька каналів зв’язку. Тому, крім адреси і електронної пошти, слід розвивати і поширювати соціальні мережі і месенджери.

Також слід додати коротку форму, яка перенаправляє на сторінку подяки. Ці форми часто настроюються, тому включають такі поля, як «Ім'я», «Місцезнаходження», «Тема», варіант прикріплення файлу і т. д.; все, що допоможе швидко зрозуміти, чому клієнт виходить на зв'язок і що необхідно зробити для швидкої допомоги користувачу.

2.8. Навігація

Навігація – це компонент веб-сайту, який допомагає людям максимально легко находити те, що вони шукають. Навігація часто відходить на другий план в дизайні сайту, тому що, це не дуже цікаво для користувача — так рахує більшість. Хоча це правда, що навігація може не викликати такого захоплення, як інші аспекти веб-архітектури. Важливість гарної навігації можна переоцінити - якщо навігація незграбна або складна, вона, швидше за все, завадить роботі користувача.

- Ієрархія інформації і те, як повинні бути названі і розташовані елементи вашого сайту малого бізнесу.
- Інформаційні та функціональні модулі
- Як організувати інформацію за важливістю для користувачів

Сортування карток — залучення користувачів до допомоги в побудові інформаційної архітектури вашого веб-сайту. У сеансі сортування карток можливо залучити учасників допомогти з організацією категорій і навіть з маркуванням категорій.

2.9. Карта сайту

Карта сайту - це, по суті, те, як користувач перейде з точки А в точку Б на сайті. Це візуальне уявлення про те, як буде здійснюватися навігація по сайту. Ефективна карта сайту надає найшвидший перехід між розділами сайту, без зайвих обхідних шляхів. Він буде містити всі сторінки сайту і те, як користувач потрапить на них.

Карта сайту може виглядати приблизно так:

Рис. 2.1. Карта сайту

У вигляді діаграми це виглядало б приблизно так:

Рис. 2.2. Карта сайту у вигляді діаграми

Звичайно, якщо веб-сайт буде трохи більш насиченим контентом, то його карта, швидше за все, буде трохи складніше, ніж зазначено вище.

В кінцевому підсумку карта сайту стане інструментом планування, який допоможе зрозуміти структуру сайту, надавши чітку візуальну перспективу.

Структура меню

Структура меню буде ключем до зручної навігації по сайту. Ефектне меню:

- Інтуїтивно зрозумілий і простий у використанні
- Дозволяє користувачам легко знаходити те, що їм потрібно, незалежно від їхнього запиту

На основі карти сайту в меню повинні бути перераховані всі категорії продуктів чи послуг, щоб клієнти могли легко знайти те, що їм потрібно або побачити, що доступно.

Загальні типи меню

Горизонтальні і вертикальні панелі навігації

Невеликі веб-сайти, як правило, використовують горизонтальні меню, в той час як більш корпоративні компанії часто використовують обидва.

Випадаюче меню

Велика панель, яка випадає з глобальної панелі навігації, на якій одночасно відображається безліч параметрів. Вони підходять для сайтів електронної комерції з великими списками категорій або сайтів, які рекламують велику кількість послуг. У стандартному меню вони виглядали б захаращеними.

Лепкий / фіксований

Цей тип меню залишається на увазі навіть при перегляді веб-сайту. CTA часто знаходиться в первинній панелі.

Товсті кінцеві рядки

Цей вид меню в нижній частині веб-сайту часто використовується для веб-сайтів, що містять більше контенту. Як правило, він включає посилання на конфіденційність / електронну пошту / юридичні посилання і контактні дані. Для сайтів електронної комерції вони також, як правило, включають способи оплати.

2.10. Тестування веб-сайту

Тестування – це безперервний процес. Люди і їх звички постійно змінюються, і те, що раніше працювало на вашому сайті, може перестати працювати через кілька місяців. Тому краще, щоб призначене для користувача тестування стало для вас звичною справою, а не доставляло час від часу незручності.

Нижче наведені деякі види тестування.

Інтерв'ю

Зворотній зв’язок з людьми з цільової аудиторії. Підгодовується список питань, які потрібно задати кожній людині, що відвідувала сайт після змін чи запуску сайту.

Огляди

Форма зворотного зв’язку від користувача, для опитування після завершення транзакції на веб-сайті також є хорошим методом збору даних. Це може бути менш особистим, ніж індивідуальне інтерв'ю, проте, транзакція буде «свіжа» в їх пам'яті, тому вони зможуть ефективно обговорити, що спрацювало, а що ні, в той час як респонденту може бути складно привести конкретні приклади.

Юзабіліті-тестування

Іноді найбільш ефективним є просто спостерігати за клієнтами, коли вони переглядають веб-сайт, відзначаючи, наскільки легко шукати, переміщатися або здійснювати покупки, якщо це сайт електронної комерції.

A / B тестування

A / B-тестування веб-сайту включає порівняння двох різних версій однієї і тієї ж веб-сторінки і тестування реакції користувачів на обидві. Один — це елемент управління, а інший — варіант, який в основному повинен бути таким же, але з однією невеликим зміною. Це може бути що завгодно, від заголовків до поєднання кольорів або зміни зображень. Потім зібрати дані на основі аналітики і конверсій, які покажуть, наскільки зміна впливає або не впливає.

2.11. Мобільна оптимізація

Оскільки все більше і більше інтернет-користувачів переглядають веб-сторінки за допомогою різних мобільних пристроїв, необхідно забезпечити доступ до веб-сайту за межами класичного робочого столу. У наші дні на мобільні пристрої доводиться більше користувачів Інтернету, ніж на комп'ютери, і це число постійно зростає.

За даними Experience Dynamics, 52% користувачів з меншою ймовірністю будуть взаємодіяти з компанією, якщо у них поганий веб-додаток.

Основи оптимізації під мобільні пристрої:

- Переконайтеся, що ваш сайт адаптивний і призначений для сенсорного управління
- Переконайтеся, що вас знайдуть в релевантних результатах пошуку
- Використати простий, але продуманий макет
- Розглянути можливість створення мобільного додатку

2.12. Зовнішній вигляд вашого сайту

Привабливий веб-сайт – це надійний веб-сайт. Це може бути поверхнево, може бути несправедливо, але так воно і є. Зовнішній вигляд

вашого веб-сайту є невід'ємною частиною користувацького досвіду. Люди, які шукають щось в мережі Інтернет, хочуть використовувати сайти, які:

- Прості у використанні;
- Виглядають красиво і сучасно

Хоча до цього в роботі ми в основному підкреслювали важливість розумного макета із інтуїтивно зрозумілою архітектурою, не менш важливим є такі речі, як колірний контраст, графіка, зображення, шрифт і те, що поєднання всіх цих елементів говорить про бренд. Всі ці елементи повинні бути узгоджені на всьому сайті. Використання абсолютно різних шрифтів або колірних схем при переході від сторінки до сторінки може збивати з пантелику і, в кінцевому підсумку, відштовхувати.

Важливі елементи, що характеризують не тільки сайт, а і його власника:

- Логотип компанії
- Схема кольорів
- Бренд

Створення такого може здатися трохи складним. У таких брендів, як Google, Apple і Twitter, їх логотипи прості, але запам'ятовуються.

Ми так рідко замислюємося про кольорову гаму відвідуваних веб-сайтів, але це важливіше, ніж ви думаєте. Згідно з дослідженням 2006 року по використанню кольору в маркетингу, 62-90% покупців оцінюють той чи інший продукт тільки на основі кольорів. Розглянемо Facebook чи інші соціальні мережі. Синій колір має тенденцію сильно виділятися. Згідно психології кольору, синій колір викликає почуття довіри і спокою. Це колір, який асоціюється зі спілкуванням, інтелектом і надійністю. Має сенс пов'язати ці почуття з чимось настільки ж потенційно особистим, як соціальні мережі.

Необхідно продумати, який настрій і яке повідомлення необхідно донести до користувача сайту. Хоча необхідно пам'ятати, що сприйняття

кольору людьми в кінцевому підсумку буде залежати від культури і часто зводиться до особистих вподобань. Ось деякі популярні кольори і їх загальні асоціації:

- Червоний - сміливий, хвилюючий, що попереджає.
- Фіолетовий - вишуканість, творчість, мудрість.
- Жовтий - оптимістичний, теплий, щасливий
- Помаранчевий - доброзичливий, впевнений, повний ентузіазму
- Зелений - природний, екологічний, спокійний

Коли справа доходить до вибору кольорів, рекомендується вибрати домінуючий колір бренду, пару додаткових акцентних кольорів і колір фону.

Однак узагальнити подібні речі складно, і те, що може спрацювати для одного сайту, може не спрацювати для іншого. Отже, як ми вже говорили раніше, A / B-тестування ніколи не буде зайвим.

ВИСНОВОК ДО РОЗДІЛУ 2

В даному розділі була викладена основна архітектура веб-сайтів. Майже всі вищезазначені рекомендації були використані при проектуванні власної екосистеми.

Також ми розуміємо, що тестування – це безперервний процес. Люди і їх звички постійно змінюються, і те, що раніше працювало на одному сайті, може перестати працювати через кілька місяців. Тому краще, щоб призначене для користувача тестування стало для вас звичною справою, а не доставляло час від часу незручності.

Краще, щоб призначене для користувача тестування стало для вас звичною справою, а не доставляло час від часу незручності.

Нижче наведені основні види тестування.

Інтерв'ю

Зворотній зв’язок з людьми з цільової аудиторії. Підгодовується список питань, які потрібно задати кожній людині, що відвідувала сайт після змін чи запуску сайту.

Огляди

Форма зворотного зв’язку від користувача, для опитування після завершення транзакції на веб-сайті також є хорошим методом збору даних. Це може бути менш особистим, ніж індивідуальне інтерв'ю, проте, транзакція буде «свіжа» в їх пам'яті, тому вони зможуть ефективно обговорити, що спрацювало, а що ні, в той час як респонденту може бути складно привести конкретні приклади.

РОЗДІЛ 3

ОСНОВИ PHP, JAVASCRIPT,CSS AND HTML5

3.1. Процес запиту / відповіді

На самому базовому рівні процес запиту / відповіді складається з веб-браузера, що запитує веб-сервер, щоб відправити йому веб-сторінку, і сервера, що відправляє сторінку назад. Потім браузер подбає про відображення сторінки.

Кожен крок у послідовності запиту і відповіді виглядає наступним чином:

- Ви вводите http://server.com в адресний рядок браузера.
- Ваш браузер шукає IP-адресу для server.com.
- Браузер видає запит на домашню сторінку на server.com.
- Запит проходить через Інтернет і надходить на веб-сервер server.com.
- Веб-сервер, отримавши запит, шукає веб-сторінку на своєму жорсткому диску.
- Веб-сторінка завантажується сервером і повертається браузеру.
- Ваш браузер відображає веб-сторінку.

Для середньої веб-сторінки цей процес виконується один раз для кожного об'єкта на сторінці: графіка, вбудованого відео або файлу Flash і навіть шаблону CSS. На другому кроці браузер знайшов IP-адресу server.com. У кожної машини, підключеної до Інтернету, є IP-адреса. Але зазвичай ми звертаємося до веб-серверів по імені, наприклад google.com.

Браузер звертається до додаткової інтернет-служби, званої службою доменних імен (DNS), щоб знайти пов'язану з нею IP-адресу, а потім використовує її для зв'язку з комп'ютером.

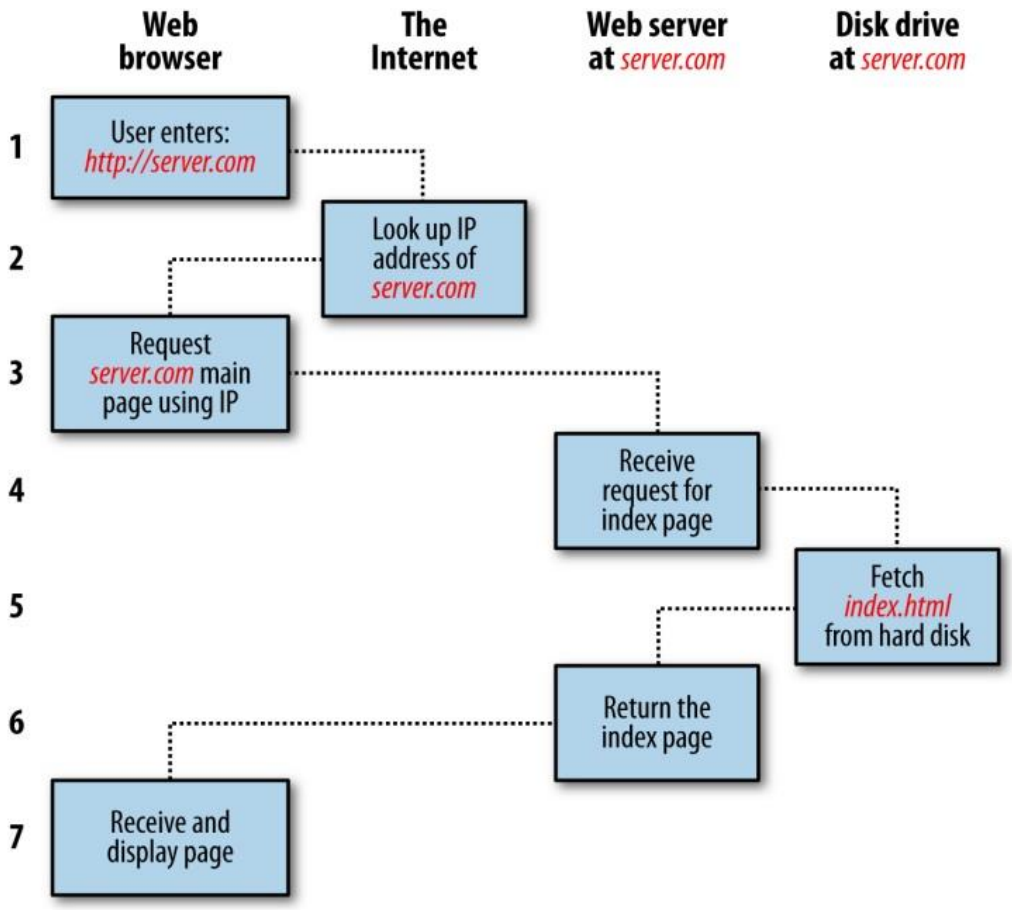

Рис. 3.1. Базова схема процесу запиту / відповіді

Для динамічних веб-сторінок процедура трохи складніше, тому що вона може об'єднати як PHP, так і MySQL.

Ось кроки для динамічної послідовності запитів / відповідей клієнт / сервер:

- Ви вводите http://server.com в адресний рядок браузера.
- Ваш браузер шукає IP-адреса для server.com.
- Браузер відправляє запит на цю адресу для домашньої сторінки веб-сервера.

- Запит проходить через Інтернет і надходить на веб-сервер server.com.
- Веб-сервер, отримавши запит, завантажує домашню сторінку зі свого жорсткого диска.
- Тепер, коли домашня сторінка знаходиться в пам'яті, веб-сервер зауважує, що це файл, який містить сценарій PHP, і передає сторінку інтерпретатора PHP.
- Інтерпретатор PHP виконує код PHP.
- Деякі команди PHP містять оператори MySQL, які інтерпретатор PHP тепер передає ядру бази даних MySQL.
- База даних MySQL повертає результати операторів назад до інтерпретатора PHP.
- Інтерпретатор PHP повертає результати виконаного коду PHP разом з результатами з бази даних MySQL на веб-сервер.
- Веб-сервер повертає сторінку клієнту, який відображає її.

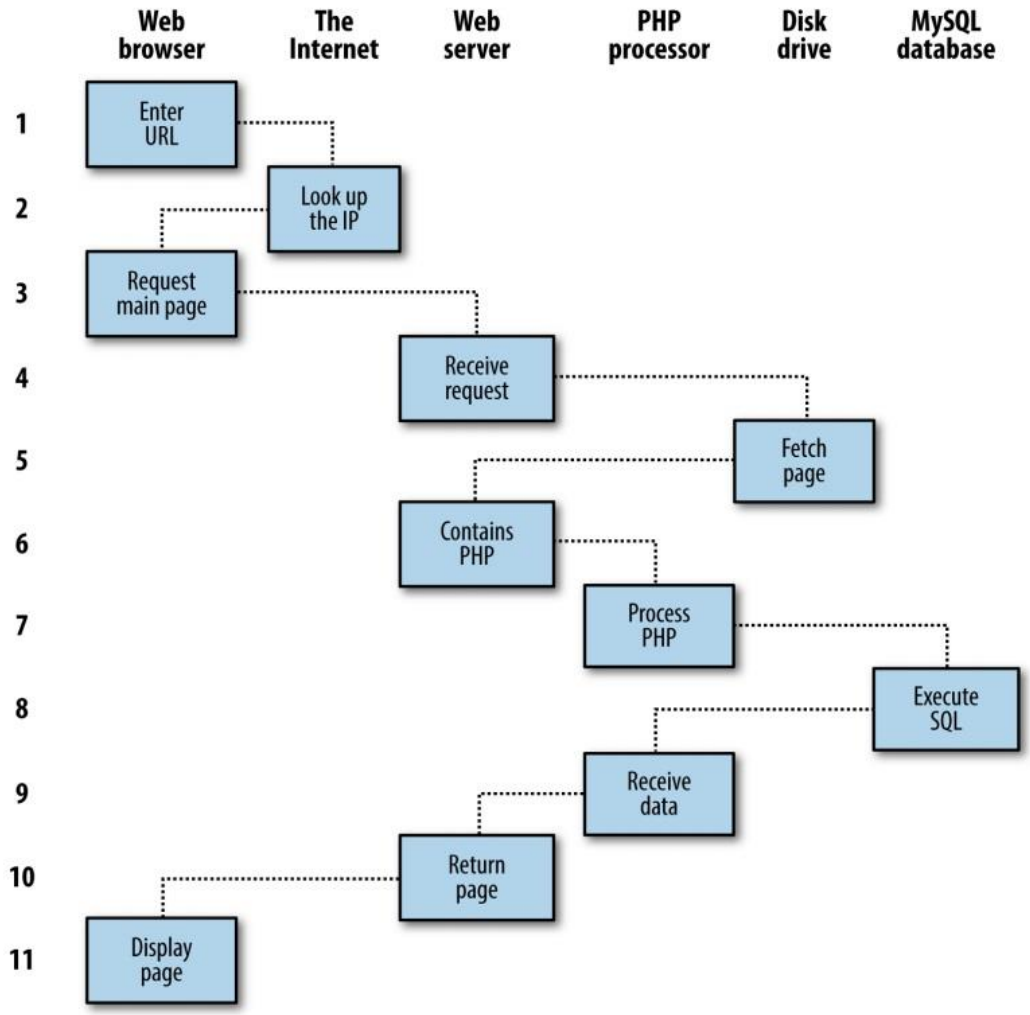

Рис. 3.2. Динамічна послідовність запитів / відповідей клієнт / сервер

Хоча корисно знати про цей процес, щоб знати, як три елементи працюють разом, на практиці не потрібно турбуватися про ці деталі, тому що всі вони відбуваються автоматично. HTML-сторінки, що повертаються браузеру в кожному прикладі, можуть містити JavaScript, який буде інтерпретуватися локально клієнтом і який може ініціювати інший запит - так само, як вбудовані об'єкти, такі як зображення.

3.2. Переваги PHP, JavaScript, CSS та HTML5

В Web версії 1.1 - це розробка таких удосконалень браузера, як Java, JavaScript, JScript (невеликий варіант JavaScript від Microsoft) і ActiveX. На стороні сервера був досягнутий прогрес в Common Gateway Interface (CGI) з

використанням мов сценаріїв, таких як Perl (альтернатива мові PHP) і сценаріїв на стороні сервера - вставка вмісту одного файлу (або виведення системи call) в інший динамічно. Коли пил влігся, три основних технології стали на голову вище за інших. Хоча Perl як і раніше залишалась популярною мовою сценаріїв і користувалась великим успіхом, простота PHP і вбудовані посилання на базу даних MySQL дозволили йому більш ніж удвічі збільшити число користувачів.

JavaScript, який став невід'ємною частиною рівняння для динамічного управління CSS (каскадних таблиць стилів) і HTML тепер взяв на себе ще більш складне завдання по роботі з клієнтською стороною процесу Ajax. У Ajax веб-сторінки виконують обробку даних і відправляють запити на веб-сервери в фоновому режимі - при цьому веб-користувач не знає, що це відбувається. Безсумнівно, симбіотична природа PHP і MySQL допомогла їм просунутися вперед, але, що в першу чергу привернуло в них розробників? Проста відповідь повинна полягати в простоті, з якою можна використовувати їх для швидкого створення динамічних елементів на веб-сайтах. MySQL - це швидка і потужна, але проста у використанні система баз даних, яка пропонує практично все, що може знадобитися веб-сайту для пошуку і надання даних браузерам. Коли PHP об'єднується з MySQL для зберігання та вилучення цих даних, ми отримуємо фундаментальні частини, необхідні для розробки сайтів соціальних мереж і зародження Web 2.0. А якщо додати в суміш JavaScript і CSS, з'являється можливіть створення високодинамічних та інтерактивних веб-сайтів.

3.3. Використання PHP

За допомогою PHP вбудувати динамічну активність в веб-сторінки дуже просто. Коли ви даєте сторінкам розширення .php, вони отримують миттєвий доступ до мови сценаріїв. З точки зору розробника, все, що потрібно зробити, це написати наступний код:

```
<?php
echo " Today is " . date("l") . ". ";
?>
```

Відкриваючий тег <?php повідомляє веб-серверу: “дозволити програмі PHP інтерпретувати весь наступний код до тега ?>”.

PHP - гнучка мова, і деякі люди вважають за краще розміщувати конструкцію PHP безпосередньо поруч з кодом PHP, наприклад:

Сьогодні <? Php echo date ("l"); ?>. Ось останні новини.

Існують також інші способи форматування і виведення інформації. Справа в тому, що з PHP у веб-розробників є мова сценаріїв, яка, хоча і не така швидка, як компіляція коду на C або аналогічною мовою, неймовірно швидкий, а також вона легко інтегрується з розміткою HTML.

Використовуючи PHP, користувач отримує необмежений контроль над своїм веб-сервером. Якщо потрібно змінити HTML на льоту, обробити кредитну карту, додати дані користувача в базу даних або отримати інформацію зі стороннього веб-сайту, можна зробити все це з тих же файлів PHP, в яких знаходиться сам HTML.

3.4. Використання JavaScript

Найстаріша з трьох основних технологій в цьому розділі, JavaScript, була створена для забезпечення доступу за допомогою сценаріїв до всіх елементів HTML-документа. Іншими словами, він забезпечує засоби для динамічної взаємодії з користувачем, таких як перевірка дійсності адреси електронної пошти в формах вводу, відображення таких запитів, як «Ви дійсно це мали на увазі?» І т. Д.

У поєднанні з CSS JavaScript - це міць динамічних веб-сторінок, які змінюються на ваших очах, а не тоді, коли сервер повертає нову сторінку.

Однак використання JavaScript також може бути складним через деякі істотні відмінності в способах його реалізації різними розробниками браузерів. В основному це відбулося, коли деякі розробники намагалися включити додаткові функції в свої браузери за рахунок сумісності зі своїми конкурентами.

На щастя, розробники в основному схаменулися і усвідомили необхідність повної сумісності між собою, тому їм не потрібно писати код з безліччю винятків. Але залишаються мільйони застарілих браузерів, які будуть використовуватися ще довгі роки. На щастя, існують рішення проблем несумісності.

Як можна використовувати базовий JavaScript, підтримуваний всіма браузерами:

```
<script type="text/javascript">
 document.write("Today is " + Date() );
</script>
```

Цей фрагмент коду вказує веб-браузеру інтерпретувати все в тегах сценарію як JavaScript, що браузер потім робить, записуючи текст Today is в поточний документ разом з датою за допомогою функції JavaScript Date(). Результат буде приблизно таким:

Today is Sun Jan 01 2020 01:23:45

JavaScript спочатку був розроблений для забезпечення динамічного контролю над різними елементами в документі HTML, і це досі є його основним застосуванням. Але все більше і більше JavaScript використовується для Ajax. Це термін для процесу доступу до веб-сервера у фоновому режимі. (Спочатку це означало «Асинхронний JavaScript і XML», але ця фраза вже трохи застаріла.)

Ajax - це основний процес, що лежить в основі того, що зараз відомо як Web 2.0, в якому веб-сторінки почали нагадувати автономні програми,

тому що вони не повинні бути повністю перезавантажуватись. Замість цього швидкий виклик Ajax може підтягнути і оновити один елемент на веб-сторінці, наприклад змінити фотографію в соціальній мережі або замінити кнопку на відповідь на питання.

3.5. Використання CSS

З появою в останні роки стандарту CSS3, CSS тепер пропонує рівень динамічної інтерактивності, раніше підтримуваний тільки JavaScript. Наприклад, ви можете не тільки стилізувати будь-який елемент HTML, щоб змінити його розміри, кольори, межі, інтервал і т.д., Але тепер ви також можете додавати анімовані переходи і перетворення на свої веб-сторінки, використовуючи всього кілька рядків CSS.

Використання CSS може бути таким же простим, як вставка декількох правил між тегами <style> і </style> в заголовку веб-сторінки, наприклад:

```
<style>
 p {
 text-align:justify;
 font-family:Helvetica;
 }
</style>
```

Ці правила змінять вирівнювання тексту за замовчуванням для тега <p>, так що абзаци, які містяться в ньому, будуть повністю вирівняні і будуть використовувати шрифт Helvetica.

Є багато різних способів розмітки правил CSS, і їх також можна включати безпосередньо в теги або зберігати набір правил в зовнішньому файлі для окремого завантаження. Ця гнучкість не тільки дозволяє точно стилізувати HTML-код, але також, наприклад, можна надавати вбудовані функції для анімації об'єктів при наведенні курсору миші на них.

3.6. Використання HTML5

Щоб вивести Інтернет за межі Web 2.0 і перейти до його наступної ітерації, був створений новий стандарт HTML для усунення всіх недоліків. Він називався HTML5 і його розробка почалась ще в 2004 році, коли був складений перший проект Mozilla Foundation і Opera Software (розробниками двох популярних веб-браузерів). Але тільки на початку 2013 року остаточний проект був представлений Консорціуму World Wide Web (W3C), міжнародному керівному органу веб-стандартів.

Базовий HTML5 - це новий стандарт, над яким веб-розробники тепер повинні працювати, і він буде залишатися в силі ще багато років.

Насправді в HTML є багато нового (і досить багато речей, які були змінені або видалені), але в підсумку ось що ми отримали:

Розмітка

Включаючи нові елементи, такі як <nav> і <footer>, і застарілі елементи, такі як <font> і .

Нові API

Наприклад, елемент <canvas> для написання і малювання на графічному полотні, елементи <audio> і <video>, автономні веб-додатки, мікродані і локальне сховище.

Додатки

Включаючи дві нові технології візуалізації: MathML (Math Markup Language) для відображення математичних формул) і SVG (Scalable Vector Graphics) для створення графічних елементів поза нового елемента <canvas>.

3.7. Веб-сервер Apache

Apache обслуговує не тільки файли HTML - він обробляє широкий спектр файлів від зображень і файлів Flash до аудіофайлів MP3, каналів RSS

(Really Simple Syndication) і так далі. Для цього кожен елемент, з яким веб-клієнт стикається на HTML-сторінці, також запитується у сервера, який потім обслуговує його.

Але ці об'єкти не обов'язково повинні бути статичними файлами, такими як зображення GIF. Всі вони можуть бути створені такими програмами, як скрипти PHP. Правильно: PHP може навіть створювати зображення та інші файли на льоту або заздалегідь, щоб їх можна було використовувати пізніше.

Для цього зазвичай є модулі, попередньо скомпільовані в Apache або PHP або викликані під час виконання. Одним з таких модулів є бібліотека GD (Graphics Draw), яку PHP використовує для створення і обробки графіки.

Apache також підтримує величезну кількість власних модулів. На додаток до модуля PHP найбільш важливими є модулі, що забезпечують безпеку. Іншими прикладами є модуль Rewrite, який дозволяє веб-серверу обробляти різний діапазон типів URL-адрес і перезаписувати їх відповідно до власних внутрішніх вимог, і модуль Proxy, який можна використовувати для обслуговування часто запитуваних сторінок з кеша для полегшення навантаження на сервер.

3.8. Вступ до PHP

У виробництві веб-сторінки будуть являти собою комбінацію PHP, HTML і JavaScript, а також деяких операторів MySQL, складених з використанням CSS і, можливо, з використанням різних елементів HTML5. Крім того, кожна сторінка може вести на інші сторінки, щоб надати користувачам можливість переходити за посиланнями і заповнювати форми. Однак можна уникнути всієї цієї складності при вивченні кожної мови.

Включення PHP в HTML

За замовчуванням документи PHP закінчуються розширенням .php. Коли веб-сервер зустрічає це розширення в запрошеному файлі, він

автоматично передає його процесору PHP. Звичайно, веб-сервери легко налаштовуються, і деякі веб-розробники вважають за краще примусово обробляти файли, що закінчуються на .htm або .html, процесором PHP, зазвичай тому, що вони хочуть приховати той факт, що вони використовують PHP.

Програма PHP відповідає за передачу чистого файлу, придатного для відображення в веб-браузері. У найпростішому випадку документ PHP виводить тільки HTML. Щоб довести це, можна взяти будь-який звичайний HTML-документ, такий як файл index.html, і зберегти його як index.php, і він буде відображатися ідентично оригіналу.

Щоб запускати команди PHP, потрібно включати тег <?php і ?>.

Цей тег можна використовувати досить гнучко. Деякі програмісти відкривають тег на початку документа і закривають його в самому кінці, виводячи HTML безпосередньо з команд PHP.

Інші, однак, вважають за краще вставляти в ці теги тільки мінімально можливі фрагменти PHP, де потрібно динамічне створення сценаріїв, залишаючи решту документа в стандартному HTML.

Другий тип програмістів зазвичай стверджує, що їх стиль кодування призводить до більш швидкого виконання коду, в той час як перший стверджує, що збільшення швидкості настільки мінімально, що не виправдовує додаткову складність багаторазового включення і виключення PHP в одному документі.

Взагалі синтаксис PHP може бути різним. Якщо шукати в Інтернеті приклади PHP, можна також зустріти код, в якому синтаксис відкриття і закриття виглядає наступним чином:

```
<?
echo "Hello world";
?>
```

Хоча не так очевидно, що викликається синтаксичний аналізатор PHP, це допустимий альтернативний синтаксис, який також зазвичай працює, але його не слід заохочувати, так як він не сумісний з XML і його використання тепер не рекомендується і може бути видалений в майбутніх версіях.

Використання коментарів в PHP

Є два способи додавання коментарів до PHP-коду. Перший перетворює окремий рядок в коментар, додаючи перед ним пару косих рис, наприклад:

```
// Comment
```

Ця версія функції коментарів - відмінний спосіб тимчасово видалити рядок коду з програми, яка видає помилки. Наприклад, можна використовувати такий коментар, щоб приховати відладочний рядок коду, поки він не знадобиться, наприклад:

```
// echo "X equals $x";
```

Другий спосіб заключається в використанні пари символів /* і */ для відкриття і закриття коментарів практично в будь-якому місці коду. Більшість, якщо не всі, програмісти використовують цю конструкцію для тимчасового коментування цілих ділянок коду, які не працюють або які з тієї чи іншої причини не хочуть інтерпретувати.

Масиви

Масиви можна уявити як кілька склеєних разом сірникових коробок. Наприклад, припустимо, що потрібно зберегти імена гравців футбольної команди з п'яти чоловік в масиві під назвою $team. Для цього можна приклеїти п'ять сірникових коробок поруч і записати імена всіх гравців на окремих аркушах паперу, помістивши по одному в кожну сірникову коробку.

Еквівалентом цього в PHP буде:

```
$team = array('Леонід', 'Антон', 'Євген', 'Олексій', 'Григорій');
```

Двомірні масиви

З масивами можна робити набагато більше. Наприклад, замість одновимірних рядів сірникових коробок вони можуть бути двовимірними матрицями або навіть мати три або більше вимірювань.

Як приклад двовимірного масиву припустимо, що потрібно відстежувати гру в хрестики-нулики, для якої потрібна структура даних з дев'яти осередків, розташованих в квадраті 3 × 3. Щоб уявити це за допомогою сірникових коробок, уявіть, що дев'ять з них приклеєні один до одного в матриці з трьох рядків на три стовпці.

Тепер можна покласти аркуш паперу зі знаком «x» або «o» в правильний коробок для сірників для кожного зіграного ходу. Щоб зробити це в PHP-коді, потрібно налаштувати масив, що містить ще три масиви.

```
$oxo = array(array('x', ' ', 'o'),
        array('o', 'o', 'x'),
        array('x', 'o', ' '));
```

Усередині зовнішньої конструкції array() вкладені три конструкції array().

Щоб потім повернути третій елемент у другому рядку цього масиву, потрібно використати наступну команду PHP, яка відобразить x:

```
echo $oxo[1][2];
```

Можна підтримувати масиви з ще більшою кількістю вимірів, просто створюючи більше масивів всередині масивів.

Оператори

Оператори - це математичні, строкові, порівняльні та логічні команди, такі як плюс, мінус, множення і ділення. PHP дуже схожий на просту арифметику, наприклад, наступний оператор виводить 8:

```
echo 6 + 2;
```

Арифметичні оператори використовуються для виконання математичних завдань. Використовувати їх можна для основних чотирьох операцій (плюс, мінус, множення і ділення), а також для знаходження модуля (залишок після поділу) і для збільшення або зменшення значення.

Оператори присвоєння використовуються для присвоєння значень змінним. Вони починаються з дуже простого = і переходять до +=, -=. Оператор += додає значення праворуч до змінної зліва замість повної заміни значення зліва. Таким чином, якщо $count починається зі значення 5, інструкція:

```
$count += 1;
echo $count;
```

поверне на екран значення 6.

Оператори порівняння зазвичай використовуються всередині такої конструкції, як оператор if, в якому потрібно порівняти два елементи. Наприклад, можна дізнатися, чи досягла змінна, яку ви збільшуєте, певного значення або інша змінна менше встановленого значення.

Наприклад, в PHP умова «Якщо час пізніше 12 години вечора і раніше 14:00 - пообідаємо» може виглядати приблизно так:

```
if ($hour > 12 && $hour < 14) dolunch();
```

Тут ми розмістили набір інструкцій для фактичного переходу до обіду в функцію, яку потрібно створити пізніше, під назвою «dolunch».

Як показує попередній приклад, зазвичай ми використовуємо логічний оператор для об'єднання результатів двох операторів порівняння. Логічний оператор також може бути введений в інший логічний оператор. Як правило, якщо щось має значення ІСТИНА або БРЕХНЯ, це можна ввести в

логічний оператор. Логічний оператор приймає два введення «істина або брехня» і дає результат «істина або брехня».

Призначення змінних

Синтаксис для присвоєння значення змінної завжди такий: змінна = значення. Або, щоб перепризначити значення іншої змінної, це інша змінна = змінна.

Є також пара інших операторів присвоювання, які можуть стати в нагоді, наприклад:

$x += 10;

який повідомляє синтаксичному аналізатору PHP додати значення праворуч (в даному випадку значення 10) до змінної $x. Точно так само можна відняти наступним чином:

$y −= 10;

Інкремент і декремент

Додавання або віднімання 1 - настільки поширена операція, що PHP надає для неї спеціальні оператори. Замість операторів += і -= можна використовувати одне з наступних:

++$x;

−−$y;

У поєднанні з тестом (оператором if) можна використовувати наступний код:

if (++$x == 10) echo $x;

Це говорить PHP спочатку збільшити значення $x, а потім перевірити, чи має воно значення 10; якщо так, вивести його значення. Також можна

вимагати від PHP збільшити (або, в наступному прикладі, зменшити) змінну після перевірки значення, наприклад:

```
if ($y−− == 0) echo $y;
```

що дає дещо інший результат. Припустимо, що $y починається з 0 перед виконанням оператора. Порівняння поверне ІСТИННИЙ результат, але після порівняння $y буде встановлено значення -1. Отже, оператор echo поверне -1.

Коротше кажучи, збільшується або зменшується змінна до або після тестування, залежить від того, де розміщений оператор збільшення або зменшення змінної.

Конкатенація рядків

Об'єднанні рядків (.) використовується для додавання одного рядка символів до іншої. Найпростіший спосіб зробити це:

```
echo "You have " . $msgs . " messages.";
```

Припускаючи, що змінна $msgs встановлена на значення 5, висновок цього рядка коду буде:

```
You have 5 messages.
```

Так само, як додається значення до числової змінної за допомогою оператора +=, можна приєднати один рядок до іншого, використовуючи .= наступним чином:

```
$bulletin .= $newsflash;
```

В цьому випадку, якщо $bulletin містить бюлетень новин, а $newsflash має флеш-повідомлення, команда додає флеш-повідомлення до бюлетеню новин, так що тепер $bulletin містить обидві рядки тексту.

3.9. Вибір технологій, платформи та мови програмування для реалізації задачі

Оскільки постановка задачі не вимагає важких математичних обчислень, а ціль роботи створити сайт з швидким розрахунком невеликого обсягу даних та можливість роботи на різних системах (десктоп та мобільний пристрій) було обрано технології PHP та JavaScript для простої обробки даних.

Мета проекту - створення потужної клієнтської та партнерської бази. Основна ціль даного проекту - є залучення більшої кількості водіїв до спільної співпраці і клієнтів до користування саме даного сервісу. Реалізація вище викладених цілей потребує простого і зрозумілого інтерфейсу для швидкого доступу до інформації про проект, збір контактних даних клієнтів та надання швидкого зворотнього зв'язку. Для цього було визначено такі етапи розробки та основні принципи дизайну:

- Створення макету дизайну сайту для клієнтоорієнтованості з урахуванням вищезазначених факторів.
- Верстка веб-застосунку на основі створеного макету, дизайнерської думки і технічної задачі.
- Програмування серверної частини для обробки отриманих даних.
- Наповнення контентом сторінок.
- Тестування релізної версії веб-застосунку для виявлення недоліків і помилок створених під час розробки.

3.10. Створення макету дизайну сайту

Перед початком верстки веб-додатку та розробки серверної частини був створений дизайн сторінки для подальшої роботи.

В першу чергу був проаналізований цільовий ринок пропозицій послуг пасажирських перевезень. На даному етапі стояв вибір між веб-додатком та мобільним додатком, але охоплення більшої аудиторії було

вирішено створити адаптивний веб-додаток. Таким чином при створенні макету сайту необхідно було враховувати і оптимізацію для мобільних пристроїв.

Згідно з дослідженням 2006 року по використанню кольору в маркетингу, 62-90% покупців оцінюють той чи інший продукт тільки на основі кольорів. Тому для кращого сприйняття додатку був продуманий дизайн сторінки в зеленому кольорі, так як відомо, що саме цей колір заспокоює очі і не створює дискомфорту для користувача. А зрозумівши, що статичний задній фон в одній кольоровій палітрі не буде викликати задоволення при користуванні я прийняв рішення примінити динамічний задній фон із трьома варіантами зображень (рис. 5.1 - 5.3).

Також, щоб додаток відчувався сучасним, фон був закріплений. Таким чином, при прокрутці, користувач бачить тільки зміну контенту, що сприяє для його освоєння.

Рис. 3.1. Фотокартка заднього фону

Рис. 3.2. Фотокартка заднього фону

Рис. 3.3. Фотокартка заднього фону

Оскільки додаток створюється як для клієнтів, так для партнерів — при відкритті сторінки користувач може обрати необхідний йому напрямок.

Враховуючи вищезазначене був розроблений макет, яким я керувався при розробці (Рис. 3.4).

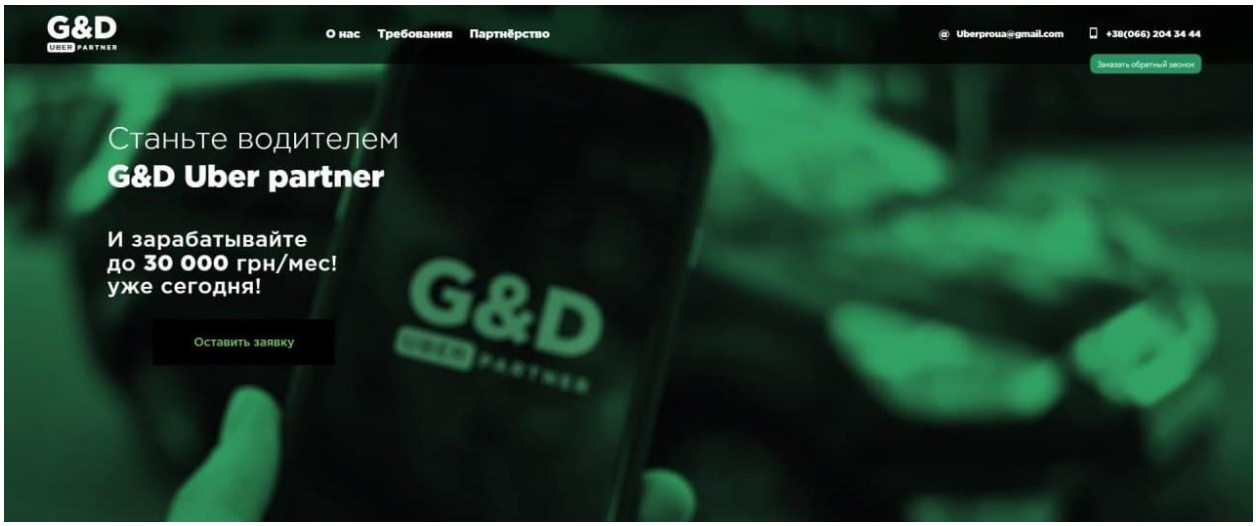

Рис. 3.4. Фінальний макет веб-додатку

3.11.Верстка веб-застосунку

Наступним кроком була сама реалізація макету. Для простоти освоєння я розділив цей етап на частини:

3.11.1. Розділення сторінки на необхідні контейнери (розмітка сторінки).

Додаток поділений на три основні контейнери: head, body, footer.

В контейнері head реалізована навігація по сайту, яка прив’язана до сторінки.

Основна частина body містить в собі наповнення сайту, а саме:

- основна інформація для користувача;
- форма зворотнього зв’язку;
- контактна інформація.

Блок footer застосовується для відображення авторських прав і посилань на інформаційні ресурси.

3.11.2. Створення графічного відображення контенту.

Для сучасного вигляду сайту була створена плавна анімація блоків з інформацією та контактною формою зворотнього зв’язку.

Як зазначалось раніше, основна кольорова палітра основана на зеленому кольорі та його відтінках. Для зручного сприйняття інформації було обрано білий та червоний кольори з використанням таких шрифтів:

- GothaProLig;
- GothaProMed;
- Glyphicons-halflings-regular.

3.11.3. Прив’язка контенту до відповідного йому контейнеру.

Сьогодні правильна подача інформації є важливим аспектом при сприйнятті користувачем сайту. Тому контент необхідно не тільки красиво оформити, а і правильно прив’язати до відповідного контейнеру, що був створений раніше. Саме через ці причини можна побачити, що основна інформація розміщується в лівій частині сторінки, а форма зворотнього зв’язку — справа. Оскільки меню додатку не відіграє роль першої необхідності воно було закріплено зверху.

3.11.4. Адаптація застосунку для мобільних пристроїв.

За статистикою близько 61% відвідувачів веб-сайтів використовують мобільний телефон. Саме по цій причині адаптація додатку для мобільних пристроїв є необхідністю, тому значна увага була приділена для адаптації кросплатформенності.

Всі блоки були перероблені для пристроїв з меншою роздільною здатністю враховуючи правила правильної подачі інформації (Рис. 3.5-5.7).

Рис. 3.5. Адаптивне відображення сторінки.

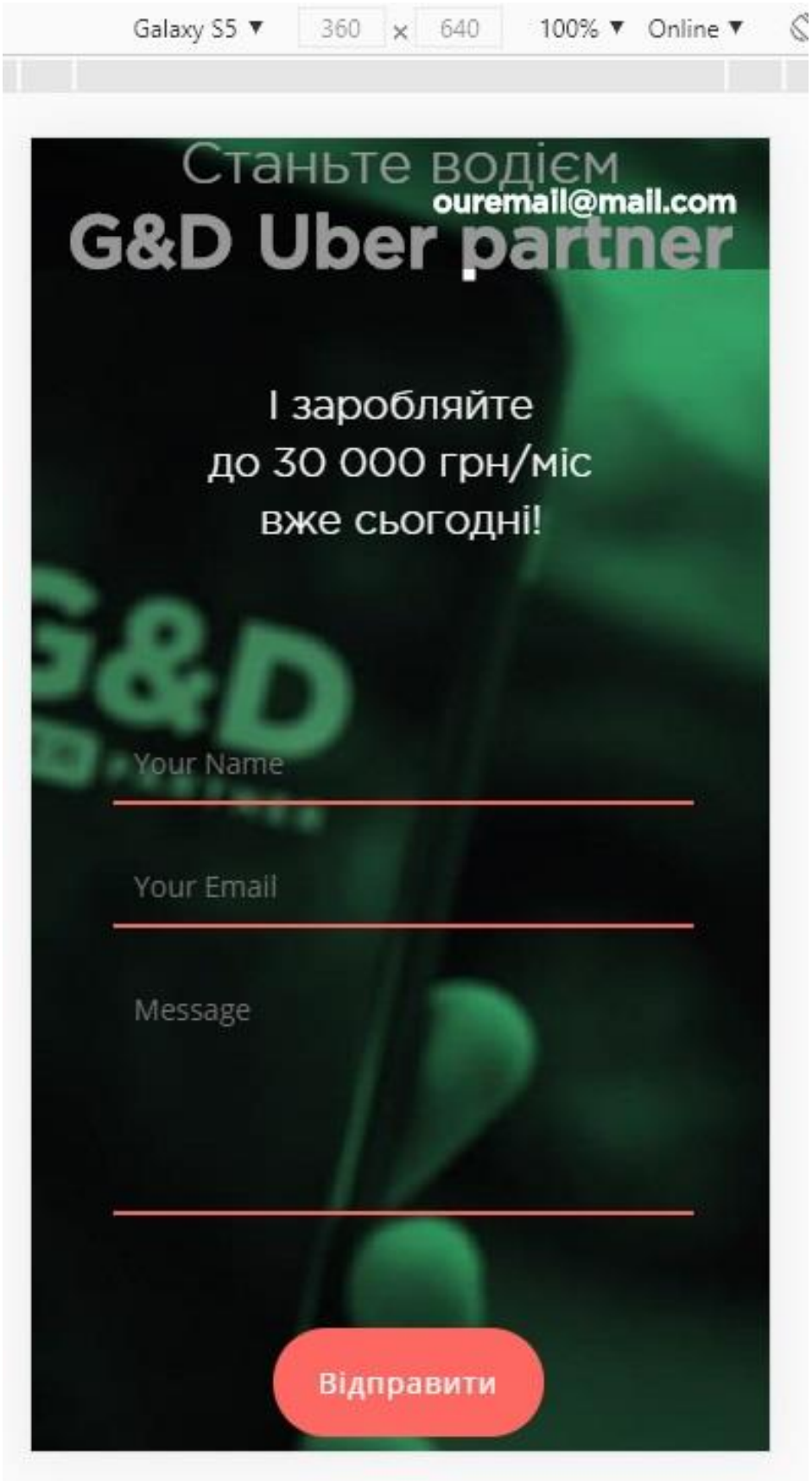

Рис. 3.6. Адаптивне відображення сторінки.

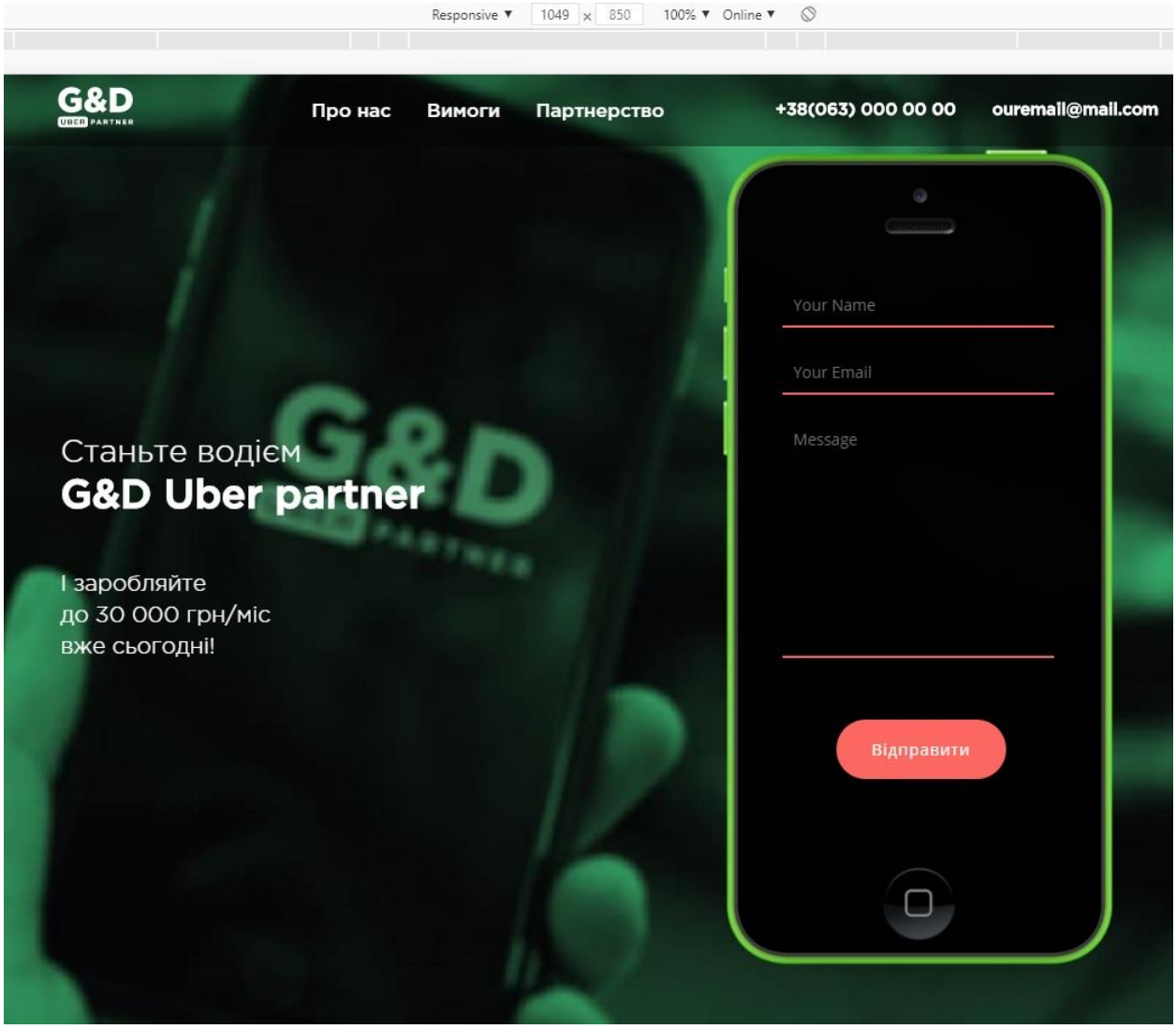

Рис. 3.7. Адаптивне відображення сторінки.

3.12. Програмування серверної частини

За обробку даних отриманих з блоку зворотнього зв’язку відповідає серверна частина.

Після того, як користувач натисне на кнопку “Відправити”, спрацює скрипт, який виконує наступні дії:

- запис даних в документ;
- відправка SMS повідомлення користувачу з підтвердженням;

3.13.Наповнення контентом сторінок

Завчасно була написана інформація, яка може знадобитись користувачу при роботі з додатком. Також була написана коротка заохочувальна реклама послуги.

3.14.Тестування релізної версії веб-застосунку

Існує два типи тестування: ручне і автоматизоване. У першому випадку контроль якості виробляє людина - тестувальник — за допомогою моделювання дій користувача. Спеціальні програми для перевірки сайту не застосовуються.

У другому випадку запуск, ініціалізація, виконання, аналіз і видача результату виробляються автоматично за допомогою спеціальних інструментів. Тестувальник обробляє отримані результати.

В даній роботі використовувалось ручне тестування веб-додатку, оскільки:

- Тестувальник оцінює продукт як звичайний користувач і може провести максимально розгорнутий тест роботи функціонала.
- Спеціаліст дивиться не тільки функціонал, але і дизайн, тому може оцінити зручність сайту.
- Можлива реалізація нетипових сценаріїв - людина може знайти баг, який не знайде програма.
- Несуттєві зміни можна перевіряти відразу після їх реалізації.

Під час тестування було знайдено ряд проблем з відображенням блоків інформації на пристроях з малим розміром екрану. Також на етапі тестування було виправлено знайдені помилки в серверній частині коду, а саме робота з даними користувача.

3.15.Робота і взаємодія з сайтом

Користувач при переході на створений сайт бачить меню з вибором напрямку співпраці. По результату натиснутої кнопки він попадає на наступну сторінку (Рис. 3.8).

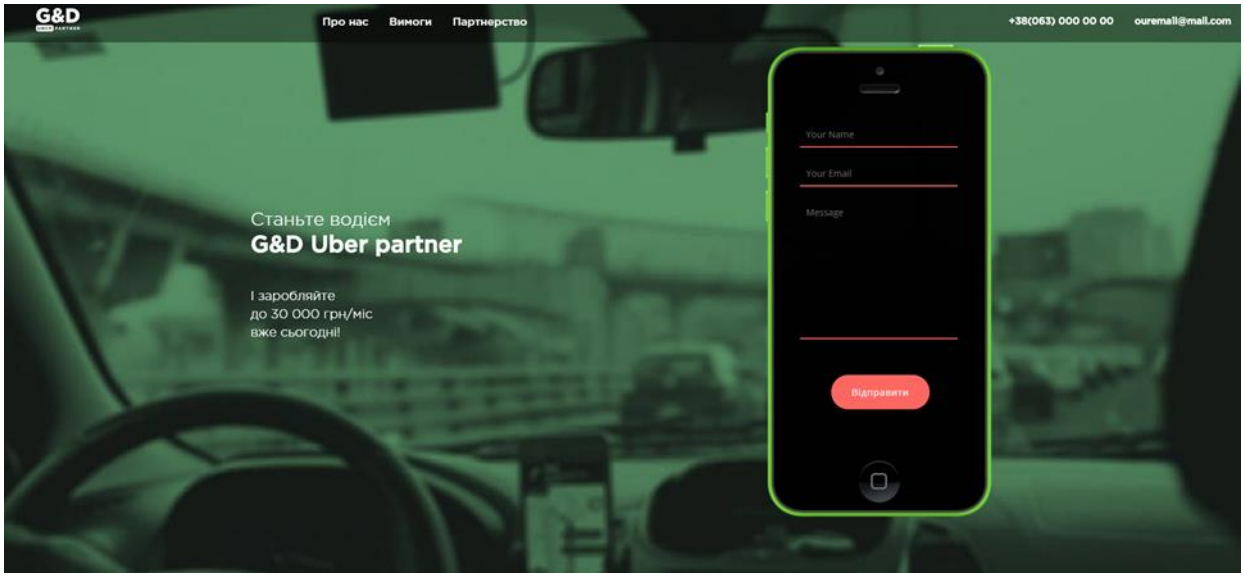

Рис. 3.8. Основна сторінка веб-додатку

На даній сторінці можна знайти основну інформацію і залишити свої дані для подальшої співпраці. А після натискання на кнопку “Відправити” користувач отримує власний Id.

В документі, куди автоматично вносяться дані партнерів, міститься вся інформація про нього, в тому числі: загальна відстань, яку він проїхав за час роботи, та сума отримана за завершенні поїздки, на основі яких проходить розрахунок бонусів (рис 5.9). Аналогічна структура є і для клієнтів.

A	B	C	D	E	F	G	H
Id	ПІБ	Email	Телефон	Повідомлення	Відстань (км)	Сума (грн)	Бонуси
1	Захаров Юрій	j.zah12121@gmail.com	38(063)152-45-87	Добрий день! Я хочу з вами співпрацювать. Маю машину Renault	97	3952	61,11340206
2							#DIV/0!
3							#DIV/0!
4							#DIV/0!
5							#DIV/0!
6							#DIV/0!
7							#DIV/0!
8							#DIV/0!
9							#DIV/0!
10							#DIV/0!
11							#DIV/0!
12							#DIV/0!
13							#DIV/0!
14							#DIV/0!
15							#DIV/0!
16							#DIV/0!
17							#DIV/0!
18							#DIV/0!
19							#DIV/0!
20							#DIV/0!
21							#DIV/0!
22							#DIV/0!
23							#DIV/0!
24							#DIV/0!
25							#DIV/0!
26							#DIV/0!
27							#DIV/0!
28							#DIV/0!
29							#DIV/0!
30							#DIV/0!
31							#DIV/0!
32							#DIV/0!
33							#DIV/0!
34							#DIV/0!

Рис. 3.9. База даних партнерів

Також було створено ряд перевірок на предмет вірно заповнених полів зворотнього зв’язку (Рис. 3.10-5.11).

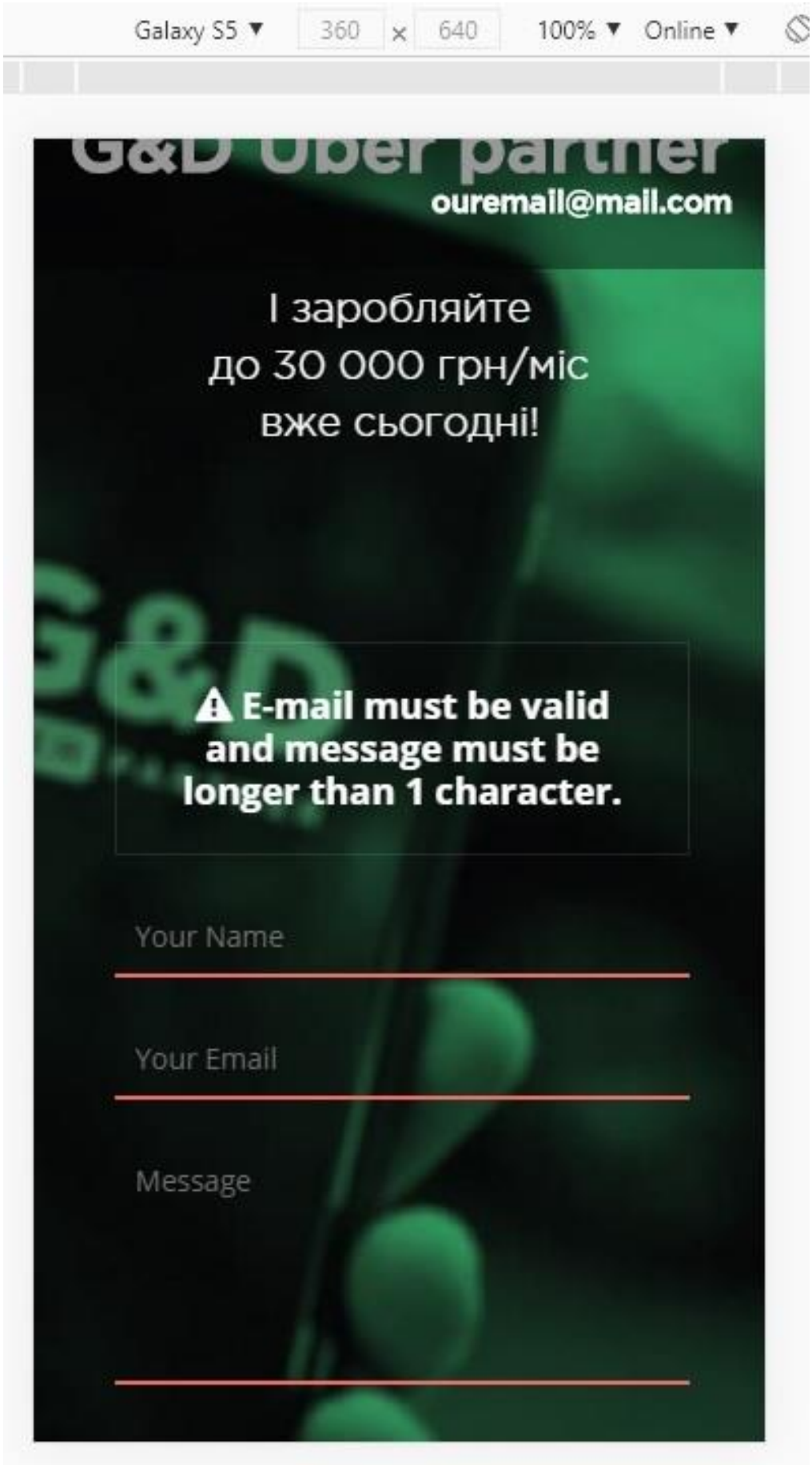

Рис. 3.10. Перевірка на вірність заповнених полів.

Рис. 3.11. Перевірка на вірність заповнених полів.

В наступних релізах веб-додатку планується реалізація додаткового функціоналу, архітектура якого уже розроблена, і перехід на іншу технологію:

- В телеграмі за допомогою боту та з використанням спеціального API буде створена система для роботи з клієнтами та партнерами, де буде використовуватись індивідуальний підхід до кожного користувача.
- Буде реалізований перехід від документу до реляційної бази даних. Оскільки відсутність бази даних в майбутньому буде негативно впливати на швидкодію екосистеми при роботі з даними.

ВИСНОВОК ДО РОЗДІЛУ 3

За допомогою PHP вбудувати динамічну активність в веб-сторінки стало набагато простіше. Коли ми даємо сторінкам розширення .php, вони отримують миттєвий доступ до мови сценаріїв.

JavaScript забезпечує засоби для динамічної взаємодії з користувачем, таких як перевірка дійсності адреси електронної пошти в формах вводу та інші.

У виробництві веб-сторінки будуть являти собою комбінацію PHP, HTML і JavaScript, складених з використанням CSS і, можливо, з використанням різних елементів HTML5. Крім того, кожна сторінка може вести на інші сторінки, щоб надати користувачам можливість переходити за посиланнями і заповнювати форми.

Коли веб-сервер зустрічає розширення .php в запрошеному файлі, він автоматично передає його процесору PHP.

ВИСНОВКИ

Оскільки постановка задачі не вимагає важких математичних обчислень, а ціль роботи створити сайт з швидким розрахунком невеликого обсягу даних та можливість роботи на різних системах (десктоп та мобільний пристрій) було обрано технології PHP та JavaScript для простої обробки даних.

Мета проекту - створення потужної клієнтської та партнерської бази. Основна ціль даного проекту - є залучення більшої кількості водіїв до спільної співпраці і клієнтів до користування саме даного сервісу. Реалізація вище викладених цілей потребує простого і зрозумілого інтерфейсу для швидкого доступу до інформації про проект, збір контактних даних клієнтів та надання швидкого зворотнього зв’язку.

Основна поставлена задача була виконана під час роботи на проектом. А під час її виконання ми побачили всі етапи створення веб-додатку, системи керування клієнтами та організації екосистеми.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Feathers JS Documentation [Електронний ресурс] – Режим доступу до ресурсу: https://docs.feathersjs.com/.
2. Современный учебник JavaScript [Електронний ресурс] – Режим доступу до ресурсу: https://learn.javascript.ru/.
3. Полное руководство по React [Електронний ресурс] – Режим доступу до ресурсу: https://learn-reactjs.ru/.
4. HTML | MDN [Електронний ресурс] – Режим доступу до ресурсу: https://developer.mozilla.org/ru/docs/Web/HTML.
5. CSS | MDN [Електронний ресурс] – Режим доступу до ресурсу: https://developer.mozilla.org/ru/docs/Web/CSS.

ДОДАТОК А

HTML

```
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width,
initial-scale=1, maximum-scale=1">
        <title>
          Diplom
          </title>
        <link rel="icon" href="images/favicon.ico">
        <link rel="apple-touch-icon" href="images/apple-touch-
icon.png">
        <link rel="apple-touch-icon" sizes="72x72"
href="images/apple-touch-icon-72x72.png">
        <link rel="apple-touch-icon" sizes="114x114"
href="images/apple-touch-icon-114x114.png">

        <link href="font-awesome/css/font-awesome.min.css"
rel="stylesheet">
        <link
href='http://fonts.googleapis.com/css?family=Open+Sans:400,700'
rel='stylesheet' type='text/css'>

        <link href="css/bootstrap.min.css" rel="stylesheet">

        <link href="css/style.css" rel="stylesheet">
    </head>

          <body>
        <div class="preloader">
            <div class="status"></div>
        </div>

        <header>
            <nav class="navbar navbar-custom navbar-top navbar-
fixed-top sticky-navigation" role="navigation">

            </nav>
        </header>
        <a href="#intro"><img src="LogoHead.svg" width="80"
class="topnavlogo"></a>
        <div class="topnav">

            <a class="" style="margin-left: 25%;"
href="#intro">РџСЂРѕ PSP°СЃ</a>
            <a href="#plans">Р’ёРјРѕРіРё</a>
            <a href="#contact">РџР°СЂС‚РЅРµСЂСЃС‚РІРѕ</a>
            <div class="contactsnew">
                <p>ouremail@mail.com</p>
```

```
                <p>+38(063) 000 00 00</p>
            </div>
        </div>
        <section id="intro">
            <div class="overlay">
                <div class="container">
                    <div class="row">

                        <div class="col-md-6 left-intro wow
fadeInLeft animated newClassForText" data-wow-offset="10" data-
wow-duration="1s">
                            <h3>РЎС,Р°PSCЊC,Рµ PIPsPґC–C”Pj</h3>
                            <h1>G&D Uber partner</h1>
                            <p style="margin-top: 10px;">Р†
Р·Р°СЂРsР±Р»СЏР№С,Рµ</p>
                            <p>РґРs 30 000 PiCЂPS/PjC–CЃ</p>
                            <p style="padding-bottom:
15px;">PIP¶Рµ CЃCЊPsPiPsPґPSC–!</p>
                        </div>
                        <div class="col-md-6">
                            <div class="mobileBg pull-right wow
fadeInRight animated" data-wow-offset="10" data-wow-
duration="1s" style="margin-top: 15px;">
                                <form class="topForm"
id="contactForm" role="form" >
                                <div class="col-sm-10">
                                    <h4 class="success">
                                        <i class="fa fa-
check"></i> Your message has been sent successfully.
                                    </h4>
                                    <h4 class="error">
                                        <i class="fa fa-
warning"></i> E-mail must be valid and message must be longer
than 1 character.
                                    </h4>
                                </div>
                                <div class="col-sm-10">
                                    <input  id="name"
type="text" name="name" placeholder="Your Name">
                                </div>
                                <div class="col-sm-10">
                                    <input  id="email"
type="email" name="email" placeholder="Your Email">
                                </div>
                                <div class="col-md-10">
                                    <textarea class="textarea-
box" id="message" rows="5" placeholder="Message"></textarea>
                                    <button style="margin-left:
30px;" class="btn-new btn-bold" type="submit" id="submit"
name="submit">P’C–PґPїCЂP°PIPёC,Pё</button>
                                </div>
                            </form>
```

```
                            <img class="img-responsive"
src="images/phones/mobile.png" alt="app" width="350" />
                            </div>
                        </div>
                    </div>
                </div>
            </div>
            <div id="supersized"></div>
        </section>


        <footer>
            <div class="container">
                <div class="col-md-12">
                    <div class="space20"></div>
                    <ul  class="social">
                        <li><a href="#"><i class="fa fa-
facebook"></i></a></li>
                        <li><a href="#"><i class="fa fa-
twitter"></i></a></li>
                        <li><a href="#"><i class="fa fa-
linkedin"></i></a></li>
                        <li><a href="#"><i class="fa fa-
dribbble"></i></a></li>
                        <li><a href="#"><i class="fa fa-
github"></i></a></li>
                    </ul>
                </div>
            </div>
        </footer>
        <div class="clearfix"></div>
        <!-- javascript Placed at the end of the document so the
pages load faster -->
        <script type="text/javascript" src="js/jquery-
1.11.2.min.js">
          </script>
        <script src="js/bootstrap.min.js">
          </script>
        <script src="js/script.js">
          </script>
    </body>
</html>
```

CSS

```
.preloader {
    position: fixed;
    top: 0;
    left: 0;
    right: 0;
    bottom: 0;
    background-color: #fefefe;
    z-index: 99999;
    height: 100%;
    width: 100%;
    overflow: hidden !important;
}

.status {
    width: 60px;
    height: 30px;
    position: absolute;
    left: 50%;
    top: 50%;
    background-image: url(../images/loader.gif);
    background-repeat: no-repeat;
    background-position: center;
    -webkit-background-size: cover;
    background-size: cover;
    margin: -15px 0 0 -30px;
}

body,html {
    height: 100%;
    width: 100%;
    max-width: 100%;
}
body {
    font-family: 'Open Sans', sans-serif;
    font-weight: 400;
    background: transparent;
    color: #333;
    overflow-x: hidden;
    margin: auto;
    background: #444;
    -webkit-font-smoothing: subpixel-antialiased;
}
h1,h2{
    font-size: 32px;
}
p{
    font-weight: 400;
    font-size: 16px;
```

```
    line-height: 1.8;
}
h1,h2,h3,h4,h5,h6 {
    font-family: 'Open Sans', sans-serif;
    font-weight: 700;
    margin-bottom: 20px;
    line-height: 1.2;
}
/* Top Navigate - start */
.topnav {
  overflow: hidden;
  position: fixed;
  background-color: rgba(0,0,0,0.4);
  width: 100%;
  z-index: 999;
  height: 64px;
}

.topnav a {
  float: left;
  color: #fff;
  font-family: "GothaProMed";
  text-align: center;
  padding: 20px 16px;
  text-decoration: none;
  font-size: 17px;
}

.topnav a:hover {
  background-color: #ddd;
  color: black;
}

.topnav a.active {
  background-color: #4CAF50;
  color: white;
}

.topnav p {
  float: right;
  color: #fff;
  font-family: "GothaProMed";
  text-align: center;
  padding: 20px 16px;
  text-decoration: none;
  font-size: 15px;
  margin: 0;
  line-height: 1.42857143;
  font-weight: 600;
  right: 1%;
}

.topnavlogo {
```

```
  margin-top: 10px;
  position: fixed;
  margin-left: 45px;
  z-index: 9999;
}

.newClassForText {

}

.newClassForText h3 {
    margin-top: 70px;
    margin-bottom: 0px;
    font-family: "GothaProLig";
    font-size: 28px;
    font-weight: 500;
}

.newClassForText h1 {
    margin-top: 0px;
    margin-bottom: 40px;
    font-family: "GothaProMed";
    font-size: 28px;
}

.newClassForText p {
    margin-top: -8px;
    margin-bottom: 0px !important;
    font-size: 20px;
    font-family: "GothaProLig";
}

.topForm {
    position: absolute;
    top: 135px;
    padding-left: 38px;
}

@media only screen and (max-width : 991px) {
    .topnavlogo {
      margin-left: -60px;
      width: 120px;
      left: 50%;
      margin-top: 100px;
      position: absolute;
    }

    .topForm {
        width: 55%;
    }

    .mobileBg {
        width: 54% !important;
```

```
    }

    #contactForm textarea {
        width: 82% !important;
    }
}

@media only screen and (max-width : 1005px) {
    .topnav a {
      display: none;
    }

    .contactsnew {
      position: absolute;
      left: 50%;
      margin-right: -50%;
      transform: translate(-50%, 0);
    }

    .topnavlogo {

    }
}
/* Top Navigate - end */
.hiddenHeader{
    display: none;
    margin: 0px;
}
.img-responsive{
    margin: 0 auto;
}
/* title Section */
.titleSection{
    text-align: center;
}
.titleSection h1,.titleSection h2{
    margin-bottom: 5px;
}
/* line */
.coloredLine{
    height: 1px;
    margin: auto;
    width: 150px;
    background: #FF6863;
    margin-top: 20px;
}
hr{
    border-color: #e4e4e4;
}
/* buttons */
.btn-new{
    background: transparent;
    border: 2px solid #FF6863;
```

```
    display: inline-block;
    padding: 15px 30px;
    border-radius: 30px;
    color: #FF6863;
    font-size: 14px;
    font-weight: 400;
    -webkit-transition: all .3s ease;
    -moz-transition: all .3s ease;
    -o-transition: all .3s ease;
    transition: all .3s ease;
    letter-spacing: 1px;
    cursor: pointer;
    line-height: 1.42857;
    text-align: center;
    white-space: nowrap;
}
.btn-new:hover,.btn-new:focus{
    background: #FC453F;
    color: #fff;
    text-decoration: none;
    outline: none;
}
.btn-bold{
    background: #FF6863;
    color: #fff;
}
.btn-bold:hover{
    background: #FC453F;
}
.light{
    font-weight: 400;
}
.bold{
    font-weight: 700;
}
.no-padding {
    padding: 0 !important;
}
.text-right{
    text-align: right;
}
/* margin spaces */
.space10 {
    margin-bottom: 10px;
}
.space20 {
    margin-bottom: 20px;
}
.space30 {
    margin-bottom: 30px;
}
.space40 {
    margin-bottom: 40px;
```

```
}
.space50 {
    margin-bottom: 50px;
}
.space60 {
    margin-bottom: 60px;
}
.space70 {
    margin-bottom: 70px;
}
.space80 {
    margin-bottom: 80px;
}
.space90 {
    margin-bottom: 90px;
}
.space100 {
    margin-bottom: 90px;
}
/* Header and nav ----------------------------------------------
----- */
.sticky-navigation {
    -webkit-transition: all .8s ease-in-out;
    -moz-transition: all .8s ease-in-out;
    -o-transition: all .8s ease-in-out;
    transition: all .8s ease-in-out;
    background: rgba(255, 255, 255, 0.9);
    border: none;
    border-bottom: 1px solid #eee;
    padding:5px 0 !important;
    margin-bottom: 0 !important;
    -webkit-backface-visibility: hidden;
    top:-100px;
}
.nav-logo{
    padding: 0px 15px;
    height: 50px;
    overflow: hidden;
}
.nav-logo:focus{
    outline: none;
}
.navbar{
    margin-bottom: 0px;
}
.navbar-nav {
    float: left;
    -webkit-transition: all .3s ease-in-out;
    -moz-transition: all .3s ease-in-out;
    -o-transition: all .3s ease-in-out;
    transition: all .3s ease-in-out;
}
```

```
.navbar-nav li a {
    padding: 10px;
    line-height: 30px;
    font-size: 14px;
    color: #444;
    position: relative;
    background: transparent !important;
    font-weight: 700;
    -webkit-transition: all .5s ease;
    -moz-transition: all .5s ease;
    -o-transition: all .5s ease;
    transition: all .5s ease;
}
.navbar-nav li a:focus{
    outline: none;
}
.navbar-nav li.active a,.navbar-nav li a:hover {
    color:#FF6863;
}
header .btn-new{
    padding: 8px 18px;
    font-size: 12px;
    margin: 6px 0px 6px 10px;
}
.navbar-toggle{
    padding: 5px 10px;
    margin-right: 10px;
}

/* Top intro ------------------------------------- */
.logo{
    margin-bottom: 40px;
}
#intro {
    position: relative;
    text-align: left;
    color: #fff;
}
#intro .overlay{
    padding: 50px 0px;
}
.left-intro{
    padding-top: 200px;
    padding-bottom: 200px;
}
#intro h1{
    font-size: 36px;
}
#intro h1 span{
    color:#FF6863;
}
#intro p{
    margin-bottom: 60px;
```

```
}
#intro .btn-new{
    margin-right: 10px;
    margin-bottom: 5px;
}
/*services------------------------------------ */
#services{
    background: #fff;
    padding: 150px 0px;
}
.servicesList{
    margin: 0px;
    padding: 0px;
    list-style: none;
}
.servicesList li{
    color: #fff;
    position: relative;
    padding: 40px;
    -webkit-transition: all .3s ease-in-out;
    -moz-transition: all .3s ease-in-out;
    -o-transition: all .3s ease-in-out;
    transition: all .3s ease-in-out;
    z-index: 2;

}
.servicesList li .number{
    position: absolute;
    top: 0px;
    left: 0px;
    font-size: 20px;
    font-weight: 700;
    width: 50px;
    height: 50px;
    line-height: 50px;
    text-align: center;
}
.servicesList li:hover{
    transform: scale(1.1,1.1);
    z-index: 99;
}
.purple{
    background: #8460a4;
}
.purple .number{
    background: #774e9c;
}
.pink{
    background: #FF6863;
}
.pink .number{
    background: #FC5653;
}
```

```
.blue{
    background: #3DCEC7;
}
.blue .number{
    background: #2ABFB5;
}
.yellow{
    background: #F4C949;
}
.yellow .number{
    background: #EABB3A;
}

#numbers .overlay{
    padding: 0px 0px 80px 0px;
}
ul.numbersList {
    margin: 0px auto;
    padding: 0px;
    overflow: hidden;
    list-style: none;
    text-align: center;
}
ul.numbersList li{
    border-right: 1px solid rgba(255, 255, 255, 0.1);
}
ul.numbersList li:last-child{
    border: none;
}
ul.numbersList h4{
    color: #FF6863;
    margin-bottom: 0px;
}
ul.numbersList span{
    font-size: 90px;
    font-weight: 700;
}

.numbers-title {
    background: none repeat scroll 0 0 rgba(255, 255, 255, 0.1);
    padding: 10px 0;
    position: relative;
    z-index: 2;
}

@media only screen and (max-width : 1200px) {
    .infoDetails{width: 100%;}
    .features-left ul{margin-top: 50px;}
    .features-right ul{margin-top: 50px;}
    .video .fluid-width-video-wrapper{ min-height: 400px;}
}

/* Medium Devices, Desktops */
```

```
@media only screen and (max-width : 991px) {

    .left-intro{
        text-align: center;
        padding-bottom: 50px;
        padding-top: 50px;
    }
    .mobileBg {
        margin: 0 auto;
        float: none !important;
        width: 60%;
    }
    .welcomeText{
        text-align: center;
    }
    ul.numbersList li{
        border:none;
    }
}
/* Small Devices, Tablets */
@media only screen and (max-width : 767px) {
    .navbar-nav { width: 100%;margin: 10px 0px;}
    .navbar-collapse {
        max-height: 355px !important;
        padding-bottom: 15px;
        padding-right: 15px;
        padding-left: 15px;
        overflow-x: visible;
        border-top: 1px solid transparent;
    }
    .infoDetails span.phone{font-size: 60px;}
    .downloadNumber{font-size:60px; }
}
/* Extra Small Devices, Phones */
@media only screen and (max-width : 480px) {
    .parallaxBg {
        background-attachment: inherit !important;
    }
    ul.numbersList h4 {font-size: 16px;}
    ul.numbersList span{font-size: 60px;}
    .infoDetails h2{font-size: 24px;}
    .infoDetails span.phone{font-size: 30px;}
    .downloadNumber{font-size:40px; }
    #intro .btn-new{padding:15px 20px; }
}
@font-face {
    font-family: "GothaProMed";
    src: url("../fonts/GothaProMed.otf") format("truetype");
    font-style: normal;
    font-weight: normal;
}

@font-face {
```

```
    font-family: "GothaProLig";
    src: url("../fonts/GothaProLig.otf") format("truetype");
    font-style: normal;
    font-weight: normal;
}
```

ДОДАТОК В

JavaScript

```
jQuery(window).load(function () {
    jQuery(".status").fadeOut();
    jQuery(".preloader").delay(500).fadeOut("slow");
});

jQuery(document).ready(function ($) {
    'use strict';


    $("#owl-carousel-testimonials").owlCarousel({
        autoPlay: 2000,
        stopOnHover: true,
        navigation: false,
        paginationSpeed: 1000,
        goToFirstSpeed: 2000,
        singleItem: true,
        autoHeight: true,
        transitionStyle: "fade"
    });
    $('#numbers ul').appear(function () {
        $('#number1').animateNumber({number: 199}, 1500);
        $('#number2').animateNumber({number: 92}, 1500);
        $('#number3').animateNumber({number: 54}, 1500);
        $('#number4').animateNumber({number: 99}, 1500);
        $('#number5').animateNumber({number: 324}, 1500);
        $('#number6').animateNumber({number: 45}, 1500);
    }, {accX: 0, accY: -200});

    $.supersized({
        slide_interval: 3500,
        transition: 1,
        transition_speed: 2000,
        slide_links: 'blank',
        slides: [
            {image: 'images/slide/1.jpg'},
            {image: 'images/slide/2.jpg'},
            {image: 'images/slide/3.jpg'}
        ]
    });
});

$(document).on('submit', '#contactForm', function (e) {
        e.preventDefault();
        var name = $("#name").val();
        var email = $("#email").val();
        var message = $("#message").val();
        var dataString = 'name=' + name + '&email=' + email + '&message=' + message;
```

```
        function isValidEmail(emailAddress) {
            var pattern = new RegExp(/^((([a-z]|\d|[!#\$%&'\*\+\-\/=\?\^_`{\|}~]|[\u00A0-
\uD7FF\uF900-\uFDCF\uFDF0-\uFFEF])+(\.([a-z]|\d|[!#\$%&'\*\+\-\/=\?\^_`{\|}~]|[\u00A0-
\uD7FF\uF900-\uFDCF\uFDF0-
\uFFEF])+)*)|((\x22)((((\x20|\x09)*(\x0d\x0a))?(\x20|\x09)+)?(([\x01-\x08\x0b\x0c\x0e-
\x1f\x7f]|\x21|[\x23-\x5b]|[\x5d-\x7e]|[\u00A0-\uD7FF\uF900-\uFDCF\uFDF0-
\uFFEF])|(\\([\x01-\x09\x0b\x0c\x0d-\x7f]|[\u00A0-\uD7FF\uF900-\uFDCF\uFDF0-
\uFFEF])))*(((\x20|\x09)*(\x0d\x0a))?(\x20|\x09)+)?(\x22)))@((([a-z]|\d|[\u00A0-
\uD7FF\uF900-\uFDCF\uFDF0-\uFFEF])|(([a-z]|\d|[\u00A0-\uD7FF\uF900-\uFDCF\uFDF0-
\uFFEF])([a-z]|\d|-|\.|_|~|[\u00A0-\uD7FF\uF900-\uFDCF\uFDF0-\uFFEF])*([a-z]|\d|[\u00A0-
\uD7FF\uF900-\uFDCF\uFDF0-\uFFEF])))\.)+(([a-z]|[\u00A0-\uD7FF\uF900-\uFDCF\uFDF0-
\uFFEF])|(([a-z]|[\u00A0-\uD7FF\uF900-\uFDCF\uFDF0-\uFFEF])([a-z]|\d|-|\.|_|~|[\u00A0-
\uD7FF\uF900-\uFDCF\uFDF0-\uFFEF])*([a-z]|[\u00A0-\uD7FF\uF900-\uFDCF\uFDF0-
\uFFEF])))\.?$/i);
            return pattern.test(emailAddress);
        }
        ;
        if (isValidEmail(email) && (message.length > 1) && (name.length > 1)) {
            $.ajax({
                type: "POST",
                url: "sendmail.php",
                data: dataString,
                success: function () {
                    $('.success').fadeIn(1000);
                    $('.error').fadeOut(500);
                }
            });
        } else {
            $('.error').fadeIn(1000);
            $('.success').fadeOut(500);
        }
        return false;
    });
```

PHP

```
<?php
if ( isset($_POST['email']) && isset($_POST['name']) && isset($_POST['message']) &&
filter_var($_POST['email'], FILTER_VALIDATE_EMAIL) ) {

  $test = "/(content-type|bcc:|cc:|to:)/i";
  foreach ( $_POST as $key => $val ) {
    if ( preg_match( $test, $val ) ) {
      exit;
    }
  }

  //
  mail( "test@test.com", Mix , $_POST['message'], "From:" . $_POST['email'] );
}
?>
```